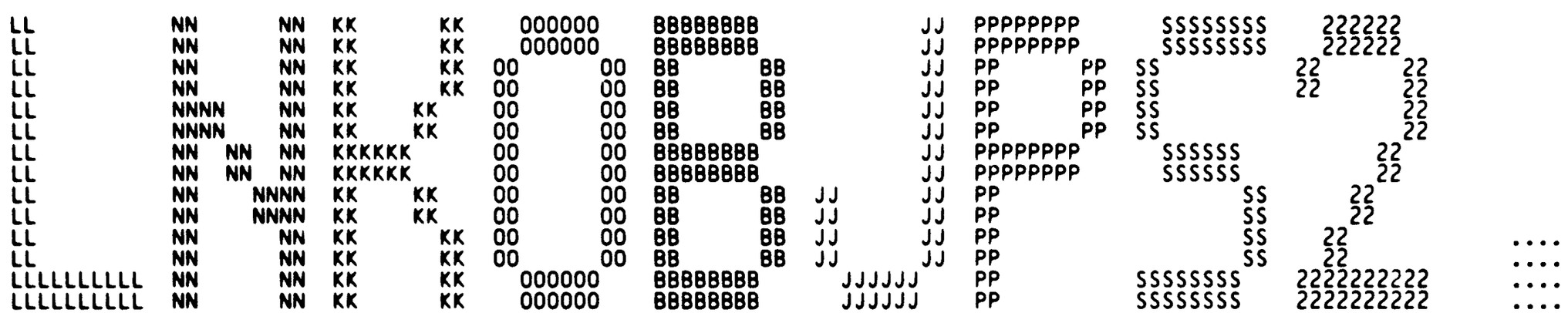
```
LLL                 IIIIIIIII     NNN         NNN  KKK         KKK  EEEEEEEEEEEEEEE   RRRRRRRRRRR
LLL                 IIIIIIIII     NNN         NNN  KKK         KKK  EEEEEEEEEEEEEEE   RRRRRRRRRRR
LLL                 IIIIIIIII     NNN         NNN  KKK         KKK  EEEEEEEEEEEEEEE   RRRRRRRRRRR
LLL                    III        NNN         NNN  KKK         KKK  EEE               RRR         RRR
LLL                    III        NNN         NNN  KKK         KKK  EEE               RRR         RRR
LLL                    III        NNN         NNN  KKK         KKK  EEE               RRR         RRR
LLL                    III        NNNNNN      NNN  KKK       KKK    EEE               RRR         RRR
LLL                    III        NNNNNN      NNN  KKK       KKK    EEE               RRR         RRR
LLL                    III        NNNNNN      NNN  KKK       KKK    EEE               RRR         RRR
LLL                    III        NNN   NNN   NNN  KKKKKKKKK        EEEEEEEEEEE       RRRRRRRRRRR
LLL                    III        NNN   NNN   NNN  KKKKKKKKK        EEEEEEEEEEE       RRRRRRRRRRR
LLL                    III        NNN   NNN   NNN  KKKKKKKKK        EEEEEEEEEEE       RRRRRRRRRRR
LLL                    III        NNN      NNNNNN  KKK       KKK    EEE               RRR   RRR
LLL                    III        NNN      NNNNNN  KKK       KKK    EEE               RRR   RRR
LLL                    III        NNN      NNNNNN  KKK       KKK    EEE               RRR   RRR
LLL                    III        NNN         NNN  KKK         KKK  EEE               RRR      RRR
LLL                    III        NNN         NNN  KKK         KKK  EEE               RRR      RRR
LLL                    III        NNN         NNN  KKK         KKK  EEE               RRR      RRR
LLLLLLLLLLLLLLLL    IIIIIIIII     NNN         NNN  KKK         KKK  EEEEEEEEEEEEEEE   RRR         RRR
LLLLLLLLLLLLLLLL    IIIIIIIII     NNN         NNN  KKK         KKK  EEEEEEEEEEEEEEE   RRR         RRR
LLLLLLLLLLLLLLLL    IIIIIIIII     NNN         NNN  KKK         KKK  EEEEEEEEEEEEEEE   RRR         RRR
```

FILEID**LNKOBJPS2



```
;  1          0001  0
;  2          0002  0 module lnk_objpass2
;  3          0003  0                 (ident = 'V04-000'
;  4          0004  0                 ,addressing_mode
;  5          0005  0                         (nonexternal = long_relative
;  6          0006  0                         ,external    = general
;  7          0007  0                         )
;  8          0008  0                 ) =
;  9          0009  0
; 10          0010  1 begin
; 11          0011  1
; 12          0012  1
; 13          0013  1 !****************************************************************************
; 14          0014  1 !*                                                                          *
; 15          0015  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
; 16          0016  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
; 17          0017  1 !*  ALL RIGHTS RESERVED.                                                     *
; 18          0018  1 !*                                                                          *
; 19          0019  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
; 20          0020  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
; 21          0021  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
; 22          0022  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
; 23          0023  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
; 24          0024  1 !*  TRANSFERRED.                                                             *
; 25          0025  1 !*                                                                          *
; 26          0026  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
; 27          0027  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
; 28          0028  1 !*  CORPORATION.                                                             *
; 29          0029  1 !*                                                                          *
; 30          0030  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
; 31          0031  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
; 32          0032  1 !*                                                                          *
; 33          0033  1 !*                                                                          *
; 34          0034  1 !****************************************************************************
; 35          0035  1 !
; 36          0036  1 !
; 37          0037  1 !
; 38          0038  1 !
; 39          0039  1 !
; 40          0040  1 !++
; 41          0041  1 !
; 42          0042  1 ! MODULE: LNK_OBJPASS2
; 43          0043  1 !
; 44          0044  1 ! FACILITY: LINKER
; 45          0045  1 !
; 46          0046  1 ! ABSTRACT: PASS 2 OF OBJECT MODULES
; 47          0047  1 !
; 48          0048  1 ! HISTORY:
; 49          0049  1 !
; 50          0050  1 !       VERSION: X01.00
; 51          0051  1 !
; 52          0052  1 !       AUTHOR: T.J. PORTER 11-APR-77
; 53          0053  1 !
; 54          0054  1 ! MODIFIED BY:
; 55          0055  1 !
; 56          0056  1 !       V03-016 ADE0002         Alan D. Eldridge        13-Aug-1984
; 57          0057  1 !               Fix call to LNK$WRTIMGBYTS so that the repeat count
```

```
;   58          0058  1 !                 is "1" instead of "0".
;   59          0059  1 !
;   60          0060  1 !       V03-015 ADE0001         Alan D. Eldridge        4-May-1984
;   61          0061  1 !               Fix access violation in STORECOMMAND.
;   62          0062  1 !
;   63          0063  1 !       V03-014 JWT0154         Jim Teague              10-Feb-1984
;   64          0064  1 !               Modules that contributed to the DST but had no psects
;   65          0065  1 !               that were other than 0-length were left out of the
;   66          0066  1 !               DMT.  Revise the philosophy here: make a DMT entry
;   67          0067  1 !               for any module that contributes to the DST, regardless
;   68          0068  1 !               of the presence or absence of non-0-length psects.
;   69          0069  1 !               I rewrote the whole mess while I was at it, placing
;   70          0070  1 !               the DMT code into a separate, heavily-commented routine,
;   71          0071  1 !               construct_dmt.
;   72          0072  1 !
;   73          0073  1 !       V03-013 JWT0136         Jim Teague              07-Sep-1983
;   74          0074  1 !               When checking for a pic shareable image after finding
;   75          0075  1 !               a non-relocatable reference, be sure to check if the
;   76          0076  1 !               image is pic AND shr.  Also, even if STDISPIC message
;   77          0077  1 !               is to be issued, it IS just a warning -- go ahead with
;   78          0078  1 !               computing the symbol reference.
;   79          0079  1 !
;   80          0080  1 !       V03-012 JWT0118         Jim Teague              04-May-1983
;   81          0081  1 !               Implement flagstack for processing symbols and
;   82          0082  1 !               expressions.
;   83          0083  1 !
;   84          0084  1 !       V03-011 JWT0116         Jim Teague              29-Apr-1983
;   85          0085  1 !               New error message LIN$_STDISPIC for STO_xD commands
;   86          0086  1 !               that store non-relocatable values into pic shr images.
;   87          0087  1 !               That displacement is computed relative to the linker's
;   88          0088  1 !               location counter (lnk$gl_locn), and once the image is
;   89          0089  1 !               relocated, that displacement will likely be wrong.
;   90          0090  1 !
;   91          0091  1 !       V03-010 JWT0094         Jim Teague              03-Feb-1983
;   92          0092  1 !               Don't write length of last DMT entry when image
;   93          0093  1 !               production is disabled.
;   94          0094  1 !
;   95          0095  1 !       V03-009 JWT0092         Jim Teague              01-Feb-1983
;   96          0096  1 !               Fix to DMT pointer adjustment when a module consists
;   97          0097  1 !               of nothing BUT 0-length psects.
;   98          0098  1 !
;   99          0099  1 !       V03-008 JWT0088         Jim Teague              12-Jan-1982
;  100          0100  1 !               Filter 0-length psects out of DMT info.
;  101          0101  1 !
;  102          0102  1 !       V03-007 JWT0084         Jim Teague              07-Jan-1982
;  103          0103  1 !               Fix some more of shareable image symbol expression
;  104          0104  1 !               problem.
;  105          0105  1 !
;  106          0106  1 !       V03-006 JWT0080         Jim Teague              15-Dec-1982
;  107          0107  1 !               Fix part of shareable image symbol relocation problem.
;  108          0108  1 !
;  109          0109  1 !       V03-005 JWT0078         Jim Teague              15-Dec-1982
;  110          0110  1 !               Change implementation of DMT.  Don't make it an image
;  111          0111  1 !               section -- let Debugger map it if needed.
;  112          0112  1 !
;  113          0113  1 !       V03-004 JWT0074         Jim Teague              09-Dec-1982
;  114          0114  1 !               Finalize Debugger DMT.
```

```
;  115          0115  1 !
;  116          0116  1 !       V03-003 JWT0061         Jim Teague              22-Oct-1982
;  117          0117  1 !               Add debugger image section to images linked /debug.
;  118          0118  1 !
;  119          0119  1 !       V03-002 JWT0041         Jim Teague              30-Jun-1982
;  120          0120  1 !               Fix for shr img symbols in based clusters
;  121          0121  1 !
;  122          0122  1 !       V03-001 JWT0038         Jim Teague              23-Jun-1982
;  123          0123  1 !               Clean up INFO#212 errors.
;  124          0124  1 !
;  125          0125  1 !
;  126          0126  1 !--
```

```
;  128     0127  1 !
;  129     0128  1 !++
;  130     0129  1 !
;  131     0130  1 ! FUNCTIONAL DESCRIPTION:
;  132     0131  1 !
;  133     0132  1 !
;  134     0133  1 !       THIS MODULE CONTAINS THE ROUTINES WHICH IMPLEMENT THE SECOND OBJECT
;  135     0134  1 !       MODULE PASS OF THE LINK. SINCE THE DATA STRUCTURES RELATING
;  136     0135  1 !       OBJECT MODULES AND FILES WERE BUILT DURING PASS 1, IT IS NOW ONLY
;  137     0136  1 !       NECESSARY TO FOLLOW THE LINKED LISTS TO ACQUIRE EACH OBJECT
;  138     0137  1 !       MODULE.
;  139     0138  1 !       PASS 2 IS CONCERNED WITH WRITING THE BINARY TO THE
;  140     0139  1 !       OUTPUT IMAGE AND PRODUCTION OF THE MODULE SYNOPSIS OF THE
;  141     0140  1 !       OUTPUT MAP LISTING. THEREFORE, THIS PASS IGNORES GSD
;  142     0141  1 !       RECORDS COMPLETELY.
;  143     0142  1 !--
;  144     0143  1 !
;  145     0144  1 library
;  146     0145  1         'LIBL32';                       ! SYSTEM DATA STRUCTURES
;  147     0146  1 require
;  148     0147  1         'PREFIX';                       ! MACROS ETC.
;  149     0262  1 library
;  150     0263  1         'DATBAS';                       ! INTERNAL DATA BASE
;  151     0264  1 require
;  152     0265  1         'ISGENC';                       ! ISECT GENERATION CONTROL TABLES
;  153     0649  1 require
;  154     0650  1         'TIRAUX';                       ! AUXILIARY MACROS FOR TIR DATA
;  155     0738  1 !
;  156     0739  1 structure
;  157     0740  1         doublevector [i,j;n,unit=4,ext=1] =
;  158     0741  1                         [2*unit*n]
;  159     0742  1                         (doublevector + 2*unit*i + unit*j) ;
;  160     0743  1 !
;  161     0744  1 linkage
;  162     0745  1         tir_proc  = jsb  : global (commandtype = 11
;  163     0746  1                                   ,commandata  = 10
;  164     0747  1                                   ,datalng     =  9
;  165     0748  1                                   ) preserve (2,3,4) nopreserve (5,6,7,8),
;  166     0749  1
;  167     0750  1         tir_proc2 = call : global (commandtype = 11
;  168     0751  1                                   ,commandata  = 10
;  169     0752  1                                   ,datalng     =  9
;  170     0753  1                                   ) ;
;  171     0754  1 forward routine
;  172     0755  1         report_illnamlng,                               ! ILLEGAL SYMBOL NAME LENGTH
;  173     0756  1         report_tirlng           : tir_proc2,            ! REPORT RECORD TOO SHORT FOR DATA REQUIRED
;  174     0757  1         report_udefpsc,                                 ! REPORT REFERENCE TO UNDEFINED PSECT
;  175     0758  1         report_useundef,                                ! REPORT USING UNDEFINED SYMBOL
;  176     0759  1         report_useudfenv,                               ! REPORT USING UNDEFINED ENVIRONMENT
;  177     0760  1         report_useudflsy,                               ! REPORT USING UNDEFINED MODULE LOCAL SYMBOL
;  178     0761  1         prohdrs,                                        ! PASS 2 PROCESSING OF HEADER RECORDS
;  179     0762  1         alloc_spcr,                                     ! ALLOCATE SPCR BLOCK
;  180     0763  1         compare_spcr,                                   ! COMPARE VALUE AND SPCR BLOCK
;  181     0764  1         compare_lit,                                    ! COMPARE VALUE AND LIT BLOCK
;  182     0765  1         alloc_lit,                                      ! ALLOCATE LIT BLOCK
;  183     0766  1         protirdbg,                                      ! PASS 2 ON TIR AND DBG RECORDS
;  184     0767  1         stackcommand            : tir_proc,             ! PROCESS TIR STACK COMMANDS
```

```
;  185     0768  1          storecommand        : tir_proc,          ! PROCESS TIR STORE COMMANDS
;  186     0769  1          operatecommand      : tir_proc,          ! PROCESS TIR OPERATE COMMANDS
;  187     0770  1          contrclcommand      : tir_proc,          ! PROCESS TIR CONTROL COMMANDS
;  188     0771  1          report_illtir       : tir_proc,          ! PROCESS ILLEGAL TIR COMMANDS
;  189     0772  1          report_illtir2      : tir_proc2,         ! PROCESS ILLEGAL TIR COMMANDS
;  190     0773  1          fndpscmapent,                            ! RETURNS MAPPING TABLE ADDRESS
;  191     0774  1          baseallimages,                           ! BASE ALL PIC,NON-BASED SHAREABLE IMAGES
;  192     0775  1          checkshrsym,                             ! VERIFY CORRECTNESS OF SHR IMG SYMBOL REF
;  193     0776  1          setaddressref,                           ! STORE INFO FOR .ADDR
;  194     0777  1          push,                                    ! PUSHES VALUES ONTO INTERNAL STACKS
;  195     0778  1          pop ;                                    ! REMOVES THEM
;  196     0779  1 !
;  197     0780  1 external routine
;  198     0781  1          crf$insrtkey,                            ! INSERT A KEY DESCRIPTION IN CREF TABLES
;  199     0782  1          crf$insrtref,                            ! INSERT A REFERENCE IN CREF TABLES
;  200     0783  1          lib$insert_tree,                         ! INSERT INTO BINARY TREE
;  201     0784  1          lib$lookup_tree,                         ! LOOKUP IN BINARY TREE
;  202     0785  1          lnk$nxtobjmod,                           ! ACQUIRE NEXT OBJECT MODULE
;  203     0786  1          lnk$nxtrec,                              ! GET NEXT RECORD OF OBJECT MODULE
;  204     0787  1          lnk$mapmhd,                              ! MAP MODULE HEADER INFORMATION
;  205     0788  1          lnk$maplng,                              ! MAP THE COMPILER NAME
;  206     0789  1          lnk$search,                              ! SYMBOL TABLE SEARCH
;  207     0790  1          lnk$searchlocal,                         ! LOCAL SYMBOL TABLE SEARCH
;  208     0791  1          lnk$fndpscnam,                           ! FIND/ENTER P-SECTION
;  209     0792  1          lnk$alloblk,                             ! MEMORY ALLOCATION ROUTINE
;  210     0793  1          lnk$getfixsize,                          ! COMPUTE SIZE OF FIXUP VECTOR
;  211     0794  1          lnk$crefixisd,                           ! CREATE FIXUP SECTION
;  212     0795  1          lnk$allovirmem,                          ! ALLOCATE MEMORY AT A SPECIFIC ADDRESS
;  213     0796  1          lnk$cluvirmem,                           ! ALLOCATE MEMORY FOR A CLUSTER
;  214     0797  1          lnk$wrtimgbyts;                          ! WRITE BYTE STREAM TO IMAGE
;  215     0798  1 !
;  216     0799  1 external literal
;  217     0800  1          lib$_normal,                             ! NORMAL RTL RETURN
;  218     0801  1          lin$_adrwrtdat,                          ! .ADDRESS IN SHR, WRT SECTION
;  219     0802  1          lin$_alredef,                            ! MULTIPLE .TRANSFER FOR A SYMBOL
;  220     0803  1          lin$_basduerrs,                          ! BASING IMAGE DUE TO REFERENCE ERRORS
;  221     0804  1          lin$_eomstk,                             ! STACK NOT COLLAPSED AT EOM
;  222     0805  1          lin$_format,                             ! ILLEGAL FORMAT
;  223     0806  1          lin$_illnamelen,                         ! ILLEGAL NAME LENGTH (SYMBOL OR PSECT)
;  224     0807  1          lin$_illrectyp,                          ! ILLEGAL RECORD TYPE
;  225     0808  1          lin$_illshropr,                          ! ILLEGAL OPERATOR ON SHR IMG SYM
;  226     0809  1          lin$_illshrxpr,                          ! TWO SHR SYMS IN EXPRESSION
;  227     0810  1          lin$_illtir,                             ! ILLEGAL TIR COMMAND
;  228     0811  1          lin$_intstkov,                           ! INTERNAL STACK OVERFLOW
;  229     0812  1          lin$_intstkun,                           ! INTERNAL STACK UNDERFLOW
;  230     0813  1          lin$_insvirmem,                          ! INSUFFICIENT VIRTUAL MEMORY
;  231     0814  1          lin$_noepm,                              ! ATTEMPT TO STACK ENTRYPOINT MASK
;  232     0815  1          lin$_nogsdref,                           ! NO GSD REF TO SHAREABLE IMAGE SYMBOL FOUND
;  233     0816  1          lin$_noimgfil,                           ! NO IMAGE FILE CREATED
;  234     0817  1          lin$_nonpicref,                          ! USE OF .LONG, ETC TO RELOCATABLE IN SHAREABLE IMAG
;  235     0818  1          lin$_notpsect,                           ! SET RELOC. BASE TO OTHER THAN PSECT CONTEXT
;  236     0819  1          lin$_rectyp,                             ! ILLEGAL RECORD
;  237     0820  1          lin$_shrpscref,                          ! INCORRECT REFERENCE TO SHAREABLE IMAGE PSECT
;  238     0821  1          lin$_shrsymref,                          ! INCORRECT REFERENCE TO SHAREABLE IMAGE SYMBOL
;  239     0822  1          lin$_shrsymtra,                          ! ATTEMPTED .TRANSFER FOR SHR IMG SYM
;  240     0823  1          lin$_stalitudf,                          ! STACK OF UNDEFINED LITERAL
;  241     0824  1          lin$_tirlng,                             ! TIR COMMAND DATA O'FLOWS RECORD
```

```
;  242          0825  1          lin$_tirnyi,                                ! UNIMPLEMEMTED TIR COMMAND
;  243          0826  1          lin$_trunc,                                 ! TRUNCATION ERRORS
;  244          0827  1          lin$_truncdat,                              ! WITH DATA OUTPUT
;  245          0828  1          lin$_udefpsc,                               ! REFERENCE TO UNDEFINED P-SECTION
;  246          0829  1          lin$_useudfenv,                             ! REFERENCE TO UNDEFINED ENVIRONMENT
;  247          0830  1          lin$_useudfen2,                             !  ...
;  248          0831  1          lin$_useudflsy,                             ! REFERENCE TO UNDEFINED MODULE-LOCAL SYMBOL
;  249          0832  1          lin$_useundef,                              ! USING AN UNDEFINED SYMBOL
;  250          0833  1          lin$_useundef2,                             !  ... IN A DEBUG / TRACEBACK RECORD
;  251          0834  1          lin$_illvps,                                ! ILLEGAL ARGUMENTS TO STORE VARIABLE POSITION/SIZE
;  252          0835  1          lin$_excpsc,                                ! TOO MANY PSECTS DEFINED (IN A SHAREABLE IMAGE BEIN
;  253          0836  1          lin$_stdispic;                              ! tir$c_sto_ld or sto_bd of scalar in a pic shr imag
;  254          0837  1 !
;  255          0838  1 external
;  256          0839  1          lnk$gl_dmtbytes,                            ! SIZE OF DMT
;  257          0840  1          lnk$gl_dmtbuffer      : vector [2],         ! BUFFER FOR ALLOCATED FOR DMT
;  258          0841  1          lnk$gw_shriscts       : word,               ! NUMBER OF ISECTS FROM PIC SHAREABLE IMAGES
;  259          0842  1          lnk$gb_matchctl       : byte,               ! GLOBAL SECTION MATCH CONTROL FOR SHAREABLE IMAGE B
;  260          0843  1          lnk$al_valctltb,                            ! CREF BY VALUE CONTROL TABLE
;  261          0844  1          lnk$al_sytblfmt,                            ! SYMBOL TABLE FORMAT DESCRIPTION BLOCK
;  262          0845  1          lnk$gl_ctlmsk         : block [,byte],      ! LINK CONTROL MASK
;  263          0846  1          lnk$gl_curisd         : ref block [,byte],  ! ADDRESS OF ISD OWNING CURRENT P-SECT
;  264          0847  1          lnk$gl_defclu         : block [,byte],      ! DEFAULT CLUSTER DESCRIPTOR
;  265          0848  1          lnk$gl_locn,                                ! CURRENT LOCATION COUNTER
;  266          0849  1          lnk$gl_dstlocn,                             ! CURRENT LOCATION IN DEBUG SYMBOL TABLE
;  267          0850  1          lnk$gl_curclu         : ref block [,byte],  ! POINTER TO CURRENT CLUSTER DESCRIPTOR
;  268          0851  1          lnk$gl_clulst,                              ! LIST HEAD OF CLUSTER LIST
;  269          0852  1          lnk$gl_fixisd         : ref block [,byte],  ! POINTER TO FIXUP ISECT DESCRIPTOR
;  270          0853  1          lnk$gl_lastgadr,                            ! LAST SPECIAL G^ ADDR ASSIGNED
;  271          0854  1          lnk$gl_record,                              ! RECORD NUMBER
;  272          0855  1          lnk$gl_rprotisd,                            ! NUMBER OF ISECTS TO REPROTECT
;  273          0856  1          lnk$gl_imgfil         : ref block [,byte],  ! IMAGE FILE FDB
;  274          0857  1          lnk$gt_imgnam         : vector [,byte],     ! IMAGE FILE NAME
;  275          0858  1          lnk$gt_symstring,                           ! ASCIC STRING OF 'SYMBOL'
;  276          0859  1          lnk$gt_pscstring,                           ! ASCIC STRING OF 'PSECT'
;  277          0860  1          lnk$gl_curomd         : ref block [,byte],  ! POINTER TO CURRENT OBJ MOD. DESCRIPTOR
;  278          0861  1          lnk$gl_curfil         : ref block [,byte] ; ! POINTER TO CURRENT FILE DESCRIPTOR
;  279          0862  1 !
;  280          0863  1 global
;  281          0864  1          lnk$gl_omdadrlst,                           ! LISTHEAD OF LIST OF OMD'S CONTAINING .ADDRESS
;  282          0865  1          lnk$gl_omdadrend,                           ! POINTER TO END OF OMDADRLST
;  283          0866  1          lnk$gl_nadromd,                             ! NUMBER OF MODULES CONTAINING .ADDRESS DATA
;  284          0867  1          lnk$gl_pshrnum        : initial (0),        ! COUNT THE NUMBER OF PSECTS
;  285          0868  1                                                      ! BOUND FOR A (POSSIBLE)
;  286          0869  1                                                      ! SHAREABLE IMAGE
;  287          0870  1          lnk$gl_spcgfix,                             ! NUMBER OF SPECIAL G^ FIXUPS
;  288          0871  1          lnk$gl_spcgimgs,                            ! NUMBER OF SHR IMGS REFERENCED WITH SPECIAL G^ FIXU
;  289          0872  1          lnk$gl_adrefs,                              ! NUMBER OF .ADDRESS FIXUPS IN LINK
;  290          0873  1          lnk$gl_shradr,                              ! NUMBER OF SHAREABLE IMAGES REF'D BY .ADDRESS
;  291          0874  1          lnk$gl_litbl,                               ! POINTER TO LITERAL TABLE
;  292          0875  1          lnk$gl_litflgs        : bitvector [lnk$c_nlits], ! DEF FLAGS FOR LITERALS
;  293          0876  1          lnk$gl_curpsc         : ref block [,byte] ; ! POINTER TO CURRENT PSECT DESCRIPTOR
;  294          0877  1 !
;  295          0878  1 global literal
;  296          0879  1          stk$c_size = 25 ;                           ! INTERNAL STACK SIZE
;  297          0880  1 !
;  298          0881  1 own
```

```
;  299     0882  1          absexpr         : initial (0),                  ! USED TO ACCUMULATE THE ABS PART OF SHR IMG EXPRESS
;  300     0883  1          shrimgsym       : ref block [,byte],            ! POINTER TO STACKED SYMBOL DESCRIPTOR IF FROM SHARE
;  301     0884  1          shrsymexpr      : ref block [,byte],            ! POINTER TO STACKED SYMBOL DESCRIPTOR IF SHR IMG SY
;  302     0885  1          obmodesc        : ref block [,byte],            ! CURRENT OBJ DESCRIPTOR
;  303     0886  1          objrecdesc      : block [dsc$c_s_bln, byte],    ! STRING DESCRIPTOR FOR OBJ RECORD
;  304     0887  1          curpsectbase,                                   ! BASE ADDRESS OF CURRENT CONTRIBUTION
;  305     0888  1          stapx           : byte,                         ! TRUE IF LAST COMMAND WAS STAPX
;  306     0889  1          stackpointer    : byte,                         ! INTERNAL STACK POINTER
;  307     0890  1          basedmsgout     : byte,                         ! TRUE IF MSG ABOUT BASING HAS BEEN ISSUED
;  308     0891  1          endrecaddr,                                     ! END OF CURRENT RECORD
;  309     0892  1          tiroffset,                                      ! OFFSET OF CURRENT COMMAND
;  310     0893  1          pdl,                                            ! POSITION DEPENDENCE LEVEL
;  311     0894  1          flags           : block [2, byte],              ! word for flags
;  312     0895  1          compustack      : vector [stk$c_size],          ! COMPUTATION STACK
;  313     0896  1          pdlstack        : vector [stk$c_size, byte],    ! PDL STACK
;  314     0897  1          psectstack      : doublevector [stk$c_size],    ! STACK TO RECORD P-SECTION CONTEXT
;  315     0898  1          shrsymstack     : vector [stk$c_size],          ! STACK SHRIMGSYM
;  316     0899  1          stapxstack      : vector [stk$c_size, byte],    ! STACK STAPX
;  317     0900  1          flagstack       : vector [stk$c_size, word],    ! word stack for symbol information
;  318     0901  1          prev_dstlen_addr,                               ! address of dstlen field of previous module
;  319     0902  1          dmtptr          : ref block [,byte];            ! current ptr into DMT
;  320     0903  1
;  321     0904  1 bind
;  322     0905  1          reclng = objrecdesc [dsc$w_length]  : word,
;  323     0906  1          objrec = objrecdesc [dsc$a_pointer] : ref block [,byte];
;  324     0907  1
;  325     0908  1          psect own = $code$;
;  326     0909  1 own
;  327     0910  1          stkdatlngtbl : vector [tir$c_maxstacod+1, byte] ! STACK COMMAND DATA LENGTH TABLE
;  328     0911  1                         initial (byte   (1,              ! 0 - STACK GLOBAL - AT LEAST ONE BYTE FOLLOWS
;  329     0912  1                                          1,              ! 1 - STACK SIGNED BYTE - ONE BYTE MUST FOLLOW
;  330     0913  1                                          2,              ! 2 - STACK SIGNED WORD - TWO BYTES MUST FOLLOW
;  331     0914  1                                          4,              ! 3 - STACK LONGWORD - FOUR BYTES MUST FOLLOW
;  332     0915  1                                          2,              ! 4 - STACK P-SECTION BASE PLUS BYTE - TWO DATA BYTE
;  333     0916  1                                          3,              ! 5 -    ''          ''       ''   ''   WORD - THREE DATA BY
;  334     0917  1                                          5,              ! 6 -    ''          ''       ''   ''   LONGWORD - FIVE DATA
;  335     0918  1                                          1,              ! 7 - STACK UNSIGNED BYTES - ONE BYTE MUST FOLLOW
;  336     0919  1                                          2,              ! 8 - STACK UNSIGNED WORD - TWO BYTES MUST FOLLOW
;  337     0920  1                                          0,              ! 9 - STACK BYTE FROM IMAGE - NO DATA BYTES
;  338     0921  1                                          0,              ! 10 - STACK WORD FROM IMAGE - NO DATA BYTES
;  339     0922  1                                          0,              ! 11 - STACK LONGWORD FROM IMAGE - NO DATA BYTES
;  340     0923  1                                          1,              ! 12 - STACK ENTRY POINT MASK - AT LEAST ONE BYTE RE
;  341     0924  1                                          2+arg$c_size,   ! 13 - STACK RESULT OF ARGUMENT CHECK.
;  342     0925  1                                          3,              ! 14 - STACK PSECT BASE PLUS BYTE (WORD PSECT NUMBER
;  343     0926  1                                          4,              ! 15 - STACK PSECT BASE PLUS WORD (WORD PSECT NUMBER
;  344     0927  1                                          6,              ! 16 - STACK PSECT BASE PLUS LONGWORD (WORD PSECT NU
;  345     0928  1                                          3,              ! 17 - STACK LOCAL SYMBOL
;  346     0929  1                                          1)),            ! 18 - STACK LITERAL
;  347     0930  1 !                                        3)),            ! 19 - STACK LOCAL ENTRY POINT
;  348     0931  1
;  349     0932  1          storectltbl : blockvector [tir$c_maxstocod-tir$c_minstocod+1,   ! THIS IS THE
;  350     0933  1                                     stoct[$c_size, byte]                 ! STORE COMMAND CONTROL TABLE
;  351     0934  1                                                                          ! EACH ENTRY OF WHICH HAS TWO BYTES, BEING
;  352     0935  1                                                                          ! THE FLAGS AND OUTPUT BYTE COUNT (RESPECTIV
;  353     0936  1                                                                          ! FOR THE CORRESPONDING STORE COMMAND.
;  354     0937  1                                                                          ! SEE 'TIRAUX' FOR FURTHER DETAILS.
;  355     0938  1                                                                          ! COMMAND TIR$C_MINSTOCOD + ...
```

```
;  356          0939  1                initial ( byte (0,1,                                    ! 0 - STORE SIGNED BYTE
;  357          0940  1                               0,2,                                     ! 1 - STORE SIGNED WORD
;  358          0941  1                               0,4,                                     ! 2 - STORE LONGWORD
;  359          0942  1                               stoctl$m_displ,1,                        ! 3 - STORE BYTE DISPLACED
;  360          0943  1                               stoctl$m_displ,2,                        ! 4 - STORE WORD DISPLACED
;  361          0944  1                               stoctl$m_displ,4,                        ! 5 - STORE LONGWORD DISPLACED
;  362          0945  1                               6^stoctl$c_mbzbit,1,                     ! 6 - STORE SHORT LITERAL
;  363          0946  1                               0,4,                                     ! 7 - STORE POSITION INDEPENDENT LONGWORD DA
;  364          0947  1                               0,5,                                     ! 8 - STORE POSITION INDEPENDENT CODE REFERE
;  365          0948  1                                                                        ! WHICH IS ADDRESSING MODE FOLLOWED BY LONGW
;  366          0949  1                               stoctl$m_rep,1,                          ! 9 - REPEATED SIGNED BYTE
;  367          0950  1                               stoctl$m_rep,2,                          ! 10 - REPEATED SIGNED WORD
;  368          0951  1                               stoctl$m_rep,4,                          ! 11 - REPEATED LONGWORD
;  369          0952  1                               0,4,                                     ! 12 - ARBITRARY BIT FIELD (UP TO 1 BYTE, BU
;  370          0953  1                               8^stoctl$c_mbzbit,1,                     ! 13 - UNSIGNED BYTE
;  371          0954  1                               16^stoctl$c_mbzbit,2,                    ! 14 - UNSIGNED WORD
;  372          0955  1                               stoctl$m_rep or (8^stoctl$c_mbzbit) ,1,  ! 15 - REPEATED UNSIGNED BYTE
;  373          0956  1                               stoctl$m_rep or (16^stoctl$c_mbzbit),2,  ! 16 - REPEATED UNSIGNED WORD
;  374          0957  1                               stoctl$m_conmbz or (8^stoctl$c_mbzbit) ,1,       ! 17 - BYTE (-128 TO +255)
;  375          0958  1                               stoctl$m_conmbz or (16^stoctl$c_mbzbit),2,       ! 18 - WORD (-32K TO + 65K)
;  376          0959  1                               stoctl$m_rep or stoctl$m_conmbz or       ! 19 - REPEATED BYTE (-128 TO +255)
;  377          0960  1                                       (8^stoctl$c_mbzbit),1,
;  378          0961  1                               stoctl$m_rep or stoctl$m_conmbz or       ! 20 - REPEATED WORD (-32K TO +65K)
;  379          0962  1                               (16^stoctl$c_mbzbit),2,                  ! 20 - REPEATED WORD (-32K TO +65K)
;  380          0963  1                               stoctl$m_rep,0,                          ! 21 - STORE REPEATED IMMEDIATE VARIABLE NO.
;  381          0964  1                               0,4));                                   ! 22 - STORE POSITION INDPENDENT RELATIVE RE
;  382          0965  1
;  383          0966  1            psect own = $own$;
;  384          0967  1  bind
;  385          0968  1            fixup_section = cstring ('Fixup section'),       ! NAME OF THE FIXUP SECTION
;  386          0969  1            objvec = objrec : ref vector [,byte],            ! NAME RECORD AS BYTE VECTOR ALSO
;  387          0970  1            rirrec = objrec : ref vector [,byte];            ! RENAME THE RECORD
;  388          0971  1
```

```
;  390          0972  1 routine report_illnamlng (symbolstring) =
;  391          0973  2 begin
;  392          0974  2 !
;  393          0975  2 ! REPORT SYMBOL NAME LENGTH ILLEGAL
;  394          0976  2 !
;  395          0977  2 map
;  396          0978  2     symbolstring : ref vector[,byte];
;  397          0979  2
;  398          0980  2 signal(lin$_illnamelen,6,lnk$gt_symstring,
;  399          0981  2          symbolstring[0],.symbolstring[0],
;  400          0982  2          sym$c_maxlng,obmodesc[omd$b_namlng],
;  401          0983  2          lnk$gl_curfil[fdb$q_filename]);
;  402          0984  2
;  403          0985  2 return false
;  404          0986  1 end;


                                                                .TITLE  LNK_OBJPASS2
                                                                .IDENT  \V04-000\

                                                                .PSECT  $CODE$,NOWRT,2

03  04  01  00  00  00  02  01  05  03  02  04  02  01  01  00000 STKDATLNGTBL:
                                                                .BYTE   1, 1, 2, 4, 2, 3, 5, 1, 2, 0, 0, 0, 1, 4, -
                                                                        3, 4, 6, 3, 1
                                                00013           .BLKB   1
00  01  30  04  02  02  02  01  02  04  00  02  00  01  00  00014 STORECTLTBL:
                                                                .BYTE   0, 1, 0, 2, 0, 4, 2, 1, 2, 2, 2, 4, 48, -
02  80  01  40  04  00  04  01  02  01  01  01  05  00  04  00023
                                                                        1, 0, 4, 0, 5, 1, 1, 1, 2, 1, 4, 0, 4, -
00  00  01  02  85  01  45  02  84  01  44  02  81  01  41  00032
                                                                        64, 1, -128, 2, 65, 1, -127, 2, 68, 1, -
                                                        04      00041
                                                                        -124, 2, 69, 1, -123, 2, 1, 0, 0, 4

                                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

                                                        0D      00000 P.AAA:    .BYTE   13
6E  6F  69  74  63  65  73  20  70  75  78  69  46      00001           .ASCII  \Fixup section\

                                                                .PSECT  $OWN$,NOEXE,2

                                                00000000        00000 ABSEXPR:  .LONG   0
                                                                00004 SHRIMGSYM:
                                                                        .BLKB   4
                                                                00008 SHRSYMEXPR:
                                                                        .BLKB   4
                                                                0000C OBMODESC:
                                                                        .BLKB   4
                                                                00010 OBJRECDESC:
                                                                        .BLKB   8
                                                                00018 CURPSECTBASE:
                                                                        .BLKB   4
                                                                0001C STAPX:    .BLKB   1
                                                                0001D STACKPOINTER:
                                                                        .BLKB   1
                                                                0001E BASEDMSGOUT:
                                                                        .BLKB   1
                                                                0001F           .BLKB   1
                                                                00020 ENDRECADDR:
```

```
                        .BLKB   4
00024 TIROFFSET:
                        .BLKB   4
00028 PDL:              .BLKB   4
0002C FLAGS:            .BLKB   2
0002E                   .BLKB   2
00030 COMPUSTACK:
                        .BLKB   100
00094 PDLSTACK:
                        .BLKB   25
000AD                   .BLKB   3
000B0 PSECTSTACK:
                        .BLKB   200
00178 SHRSYMSTACK:
                        .BLKB   100
001DC STAPXSTACK.
                        .BLKB   25
001F5                   .BLKB   3
001F8 FLAGSTACK:
                        .BLKB   50
0022A                   .BLKB   2
0022C PREV_DSTLEN_ADDR:
                        .BLKB   4
00230 DMTPTR:           .BLKB   4

                        .PSECT  $GLOBAL$,NOEXE,2

00000 LNK$GL_OMDADRLST::
                        .BLKB   4
00004 LNK$GL_OMDADREND::
                        .BLKB   4
00008 LNK$GL_NADROMD::
                        .BLKB   4
00000000  0000C LNK$GL_PSHRNUM::
                        .LONG   0
00010 LNK$GL_SPCGFIX::
                        .BLKB   4
00014 LNK$GL_SPCGIMGS::
                        .BLKB   4
00018 LNK$GL_ADREFS::
                        .BLKB   4
0001C LNK$GL_SHRADR::
                        .BLKB   4
00020 LNK$GL_LITBL::
                        .BLKB   4
00024 LNK$GL_LITFLGS::
                        .BLKB   32
00044 LNK$GL_CURPSC::
                        .BLKB   4

              ISD$C_SIZE==          88
              HDR$K_FILLCHR==       255
              IHD$K_SHR==           2
              IHD$K_ACTIVOFF==      48
              IHD$K_SYMDBGOFF==     68
              IHD$K_IMGIDOFF==      88
              IHD$K_PATCHOFF==      168
```

```
IHD$K_MAXLENGTH==    168
HDR$K_MINFILL==      2
STK$C_SIZE==         25
RECLNG=              OBJRECDESC
OBJREC=              OBJRECDESC+4
FIXUP_SECTION=       P.AAA
OBJVEC=              OBJRECDESC+4
TIRREC=              OBJRECDESC+4
                .EXTRN  CRF$INSRTKEY, CRF$INSRTREF
                .EXTRN  LIB$INSERT_TREE
                .EXTRN  LIB$LOOKUP_TREE
                .EXTRN  LNK$NXTOBJMOD, LNK$NXTREC
                .EXTRN  LNK$MAPMHD, LNK$MAPLNG
                .EXTRN  LNK$SEARCH, LNK$SEARCHLOCAL
                .EXTRN  LNK$FNDPSCNAM, LNK$ALLOBLK
                .EXTRN  LNK$GETFIXSIZE, LNK$CREFIXISD
                .EXTRN  LNK$ALLOVIRMEM, LNK$CLUVIRMEM
                .EXTRN  LNK$WRTIMGBYTS, LIB$_NORMAL
                .EXTRN  LIN$_ADRWRTDAT, LIN$_ALREDEF
                .EXTRN  LIN$_BASDUERRS, LIN$_EOMSTK
                .EXTRN  LIN$_FORMAT, LIN$_ILLNAMELEN
                .EXTRN  LIN$_ILLRECTYP, LIN$_ILLSHROPR
                .EXTRN  LIN$_ILLSHRXPR, LIN$_ILLTIR
                .EXTRN  LIN$_INTSTKOV, LIN$_INTSTKUN
                .EXTRN  LIN$_INSVIRMEM, LIN$_NOEPM
                .EXTRN  LIN$_NOGSDREF, LIN$_NOIMGFIL
                .EXTRN  LIN$_NONPICREF, LIN$_NOTPSECT
                .EXTRN  LIN$_RECTYP, LIN$_SHRPSCREF
                .EXTRN  LIN$_SHRSYMREF, LIN$_SHRSYMTRA
                .EXTRN  LIN$_STALITUDF, LIN$_TIRLNG
                .EXTRN  LIN$_TIRNYI, LIN$_TRUNC
                .EXTRN  LIN$_TRUNCDAT, LIN$_UDEFPSC
                .EXTRN  LIN$_USEUDFENV, LIN$_USEUDFEN2
                .EXTRN  LIN$_USEUDFLSY, LIN$_USEUNDEF
                .EXTRN  LIN$_USEUNDEF2, LIN$_ILLVPS
                .EXTRN  LIN$_EXCPSC, LIN$_STBISPIC
                .EXTRN  LNK$GL_DMTBYTES
                .EXTRN  LNK$GL_DMTBUFFER
                .EXTRN  LNK$GW_SHRISCTS
                .EXTRN  LNK$GB_MATCHCTL
                .EXTRN  LNK$AL_VALCTLTB
                .EXTRN  LNK$AL_SYTBLFMT
                .EXTRN  LNK$GL_CTLMSK, LNK$GL_CURISD
                .EXTRN  LNK$GL_DEFCLU, LNK$GL_LOCN
                .EXTRN  LNK$GL_DSTLOCN, LNK$GL_CURCLU
                .EXTRN  LNK$GL_CLULST, LNK$GL_FIXISD
                .EXTRN  LNK$GL_LASTGADR
                .EXTRN  LNK$GL_RECORD, LNK$GL_RPROTISD
                .EXTRN  LNK$GL_IMGFIL, LNK$GT_IMGNAM
                .EXTRN  LNK$GT_SYMSTRING
                .EXTRN  LNK$GT_PSCSTRING
                .EXTRN  LNK$GL_CUROMD, LNK$GL_CURFIL

                .PSECT  $CODE$,NOWRT,2

0000 00000 REPORT_ILLNAMLNG:
                .WORD   Save nothing                    ; 0972
```

```
                7E 00000000G   00              14   C1 00002          ADDL3   #20, LNK$GL_CURFIL, -(SP)                ; 0983
                7E 00000000'   EF              28   C1 0000A          ADDL3   #40, OBMODESC, -(SP)                     ; 0982
                                               1F   DD 00012          PUSHL   #31                                      ; 0983
                               7E        04    BC   9A 00014          MOVZBL  @SYMBOLSTRING, -(SP)
                                         04    AC   DD 00018          PUSHL   SYMBOLSTRING
                                  00000000G    00   9F 0001B          PUSHAB  LNK$GT_SYMSTRING                         ; 0980
                                               06   DD 00021          PUSHL   #6                                       ; 0983
                                  00000000G    8F   DD 00023          PUSHL   #LIN$_ILLNAMELEN
                    00000000G  00              08   FB 00029          CALLS   #8, LIB$SIGNAL
                                               50   D4 00030          CLRL    R0                                       ; 0985
                                                    04 00032          RET                                              ; 0986

; Routine Size:  51 bytes,    Routine Base:  $CODE$ + 0042
```

```
;  406          0987  1 routine report_tirlng : tir_proc2 =
;  407          0988  2 begin
;  408          0989  2 !
;  409          0990  2 ! REPORT RECORD TOO SHORT TO HOLD REQUIRED DATA
;  410          0991  2 !
;  411          0992  2 external register
;  412          0993  2     datalng = 9;
;  413          0994  2
;  414          0995  2 signal(lin$_tirlng,4,.datalng,obmodesc[omd$b_namlng],   !SIGNAL RECORD TOO SHORT
;  415          0996  2          .lnk$gl_record,lnk$gl_curfil[fdb$q_filename]);
;  416          0997  2
;  417          0998  2 return false
;  418          0999  1 end;


                                        0000 00000 REPORT_TIRLNG:
                                                   .WORD   Save nothing                                    ; 0987
                    7E 00000000G   00              14  C1 00002          ADDL3   #20, LNK$GL_CURFIL, -(SP)  ; 0996
                                   00000000G       00  DD 0000A          PUSHL   LNK$GL_RECORD              ;
                    7E 00000000'   EF              28  C1 00010          ADDL3   #40, OBMODESC, -(SP)       ; 0995
                                                   59  DD 00018          PUSHL   DATALNG                    ; 0996
                                                   04  DD 0001A          PUSHL   #4                         ;
                                   00000000G       8F  DD 0001C          PUSHL   #LIN$_TIRLNG               ;
                       00000000G   00              06  FB 00022          CALLS   #6, LIB$SIGNAL             ;
                                                   50  D4 00029          CLRL    R0                         ; 0998
                                                       04 0002B          RET                                ; 0999

; Routine Size:  44 bytes,    Routine Base:  $CODE$ + 0075
```

```
;  420          1000  1 routine report_udefpsc (psectnum) =
;  421          1001  2 begin
;  422          1002  2 !
;  423          1003  2 ! REPORT REFERENCE TO AN UNDEFINED PSECT
;  424          1004  2 !
;  425          1005  2 signal(lin$_udefpsc,3,,psectnum,obmodesc[omd$b_namlng],
;  426          1006  2         lnk$gl_curfil[fdb$q_filename]);
;  427          1007  2
;  428          1008  2 return false
;  429          1009  1 end;


                                        0000 00000 REPORT_UDEFPSC:
                                                           .WORD   Save nothing                                    ; 1000
                 7E 00000000G   00          14   C1 00002          ADDL3   #20, LNK$GL_CURFIL, -(SP)               ; 1006
                 7E 00000000'   EF          28   C1 0000A          ADDL3   #40, OBMODESC, -(SP)                    ; 1005
                                    04     AC   DD 00012          PUSHL   PSECTNUM                                ; 1006
                                           03   DD 00015          PUSHL   #3                                      ;
                                00000000G  8F   DD 00017          PUSHL   #LIN$_UDEFPSC                           ;
                    00000000G   00         05   FB 0001D          CALLS   #5, LIB$SIGNAL                          ;
                                           50   D4 00024          CLRL    R0                                      ; 1008
                                                04 00026          RET                                             ; 1009

; Routine Size:  39 bytes,    Routine Base:  $CODE$ + 00A1
```

```
;  431          1010  1 routine report_useundef (symbolname) =
;  432          1011  2 begin
;  433          1012  2 !
;  434          1013  2 ! REPORT ATTEMPT TO USE AN UNDEFINED SYMBOL
;  435          1014  2 !
;  436          1015  2 if .lnk$gl_ctlmsk [lnk$v_dbgrec]
;  437          1016  2 then signal (lin$_useundef2, 3, .symbolname
;  438          1017  2                ,obmodesc [omd$b_naming], lnk$gl_curfil [fdb$q_filename]
;  439          1018  2                )
;  440          1019  2 else signal (lin$_useundef, 5, .symbolname
;  441          1020  3                ,lnk$gl_curpsc [psc$b_naming], (.lnk$gl_locn-.curpsectbase)
;  442          1021  2                ,obmodesc [omd$b_naming], lnk$gl_curfil [fdb$q_filename]
;  443          1022  2                );
;  444          1023  2
;  445          1024  2 return false
;  446          1025  1 end;

                                        0004 00000 REPORT_USEUNDEF:
                                                   .WORD   Save R2                                          ; 1010
                    52 00000000G   00    9E 00002          MOVAB   LIB$SIGNAL, R2
 51 00000000G       00             14    C1 00009          ADDL3   #20, LNK$GL_CURFIL, R1                   ; 1017
 50 00000000'       EF             28    C1 00011          ADDL3   #40, OBMODE$C, R0
                    12 00000000G   00    E9 00019          BLBC    LNK$GL_CTLMSK+2, 1$                      ; 1015
                                   03    BB 00020          PUSHR   #^M<R0,R1>                               ; 1017
                              04   AC    DD 00022          PUSHL   SYMBOLNAME
                                   03    DD 00025          PUSHL   #3
                         00000000G 8F    DD 00027          PUSHL   #LIN$_USEUNDEF2
                              62         05    FB 0002D          CALLS   #5, LIB$SIGNAL
                                   24    11 00030          BRB     2$
                                   03    BB 00032 1$:      PUSHR   #^M<R0,R1>                               ; 1021
 7E 00000000G       00 00000000'   EF    C3 00034          SUBL3   CURPSECTBASE, LNK$GL_LOCN, -(SP)         ; 1020
 7E 00000000'       EF             2D    C1 00040          ADDL3   #45, LNK$GL_CURPSC, =(SP)
                              04   AC    DD 00048          PUSHL   SYMBOLNAME                               ; 1021
                                   05    DD 0004B          PUSHL   #5
                         00000000G 8F    DD 0004D          PUSHL   #LIN$_USEUNDEF
                              62         07    FB 00053          CALLS   #7, LIB$SIGNAL
                                   50    D4 00056 2$:      CLRL    R0                                       ; 1024
                                         04 00058          RET                                              ; 1025

; Routine Size:  89 bytes,    Routine Base:  $CODE$ + 00C8
```

```
;  448          1026  1 routine report_useudfenv (envdesc,envindex) =
;  449          1027  2 begin
;  450          1028  2 !
;  451          1029  2 ! REPORT REFERENCE TO UNDEFINED ENVIRONMENT
;  452          1030  2 !
;  453          1031  2 map
;  454          1032  2     envdesc : ref block[,byte];
;  455          1033  2
;  456          1034  2 local
;  457          1035  2     errorcode,
;  458          1036  2     arg1;
;  459          1037  2
;  460          1038  2 if .envdesc eql 0
;  461          1039  3 then begin
;  462          1040  3     errorcode = lin$_useudfen2;
;  463          1041  3     arg1 = .envindex;
;  464          1042  3     end
;  465          1043  3 else begin
;  466          1044  3     errorcode = lin$_useudfenv;
;  467          1045  3     arg1 = envdesc[nvd$b_namlng];
;  468          1046  2     end;
;  469          1047  2 signal(.errorcode,5,.arg1,
;  470          1048  2          lnk$gl_curpsc[psc$b_namlng],
;  471          1049  2          (.lnk$gl_locn-.curpsectbase),
;  472          1050  2          obmodesc[omd$b_namlng],
;  473          1051  2          lnk$gl_curfil[fdb$q_filename]);
;  474          1052  2
;  475          1053  2 return false
;  476          1054  1 end;


                                0000 00000 REPORT_USEUDFENV:
                                           .WORD   Save nothing                                    ;  1026
                          04    AC  D5 00002         TSTL    ENVDESC                                ;  1038
                                0D  12 00005         BNEQ    1$                                     ;
                51 00000000G    8F  D0 00007         MOVL    #LIN$_USEUDFEN2, ERRORCODE             ;  1040
                50        08    AC  D0 0000E         MOVL    ENVINDEX, ARG1                         ;  1041
                                0C  11 00012         BRB     2$                                     ;  1038
                51 00000000G    8F  D0 00014 1$:     MOVL    #LIN$_USEUDFENV, ERRORCODE             ;  1044
          50        04    AC      12  C1 0001B       ADDL3   #18, ENVDESC, ARG1                     ;  1045
       7E 00000000G     00          14  C1 00020 2$:  ADDL3   #20, LNK$GL_CURFIL, -(SP)            ;  1051
       7E 00000000'     EF          28  C1 00028       ADDL3   #40, OBMODESC, -(SP)                ;  1050
       7E 00000000G     00 00000000'EF  C3 00030       SUBL3   CURPSECTBASE, LNK$GL_LOCN, -(SP)    ;  1049
       7E 00000000'     EF          2D  C1 0003C       ADDL3   #45, LNK$GL_CURPSC, -(SP)           ;  1048
                                    50  DD 00044       PUSHL   ARG1                                ;  1051
                                    05  DD 00046       PUSHL   #5                                  ;
                                    51  DD 00048       PUSHL   ERRORCODE                           ;
             00000000G  00          07  FB 0004A       CALLS   #7, LIB$SIGNAL                      ;
                                    50  D4 00051       CLRL    R0                                  ;  1053
                                        04 00053       RET                                         ;  1054

; Routine Size:  84 bytes,    Routine Base:  $CODE$ + 0121
```

```
;  478          1055  1 routine report_useudflsy (symbolname) =
;  479          1056  2 begin
;  480          1057  2 !
;  481          1058  2 ! REPORT REFERENCE TO UNDEFINED LOCAL SYMBOL
;  482          1059  2 !
;  483          1060  2 signal(lin$_useudflsy,5,.symbolname,
;  484          1061  2         lnk$gl_curpsc[psc$b_naming],(.lnk$gl_locn-.curpsectbase),
;  485          1062  2                 obmodesc[omd$b_naming],lnk$gl_curfil[fdb$q_filename]);
;  486          1063  2 return false;
;  487          1064  1 end;


                                        0000 00000 REPORT_USEUDFLSY:
                                                   .WORD   Save nothing                                    ; 1055
7E 00000000G   00                 14   C1 00002          ADDL3   #20, LNK$GL_CURFIL, -(SP)                ; 1062
7E 00000000'   EF                 28   C1 0000A          ADDL3   #40, OBMODESC, -(SP)
7E 00000000G   00 00000000'       EF   C3 00012          SUBL3   CURPSECTBASE, LNK$GL_LOCN, -(SP)         ; 1061
7E 00000000'   EF                 2D   C1 0001E          ADDL3   #45, LNK$GL_CURPSC, -(SP)                ; 1062
                            04    AC   DD 00026          PUSHL   SYMBOLNAME
                                  05   DD 00029          PUSHL   #5
                       00000000G  8F   DD 0002B          PUSHL   #LIN$_USEUDFLSY
   00000000G   00                 07   FB 00031          CALLS   #7, LIB$SIGNAL
                                  50   D4 00038          CLRL    R0                                      ; 1063
                                       04 0003A          RET                                             ; 1064

; Routine Size:  59 bytes,    Routine Base:  $CODE$ + 0175
```

```
;  489          1065  1 routine report_illtir : tir_proc =
;  490          1066  2 begin
;  491          1067  2
;  492          1068  2 external register
;  493          1069  2     commandtype = 11 : byte;
;  494          1070  2
;  495          1071  2 signal(lin$_illtir,4,
;  496          1072  2          .commandtype,obmodesc[omd$b_namlng],
;  497          1073  2          .lnk$gl_record,lnk$gl_curfil[fdb$q_filename]);
;  498          1074  2
;  499          1075  2 return false
;  500          1076  1 end;

                    7E 00000000G  00              14  C1 00000 REPORT_ILLTIR:
                                                                ADDL3   #20, LNK$GL_CURFIL, -(SP)                   ; 1073
                                      00000000G   00  DD 00008          PUSHL   LNK$GL_RECORD
                    7E 00000000'  EF              28  C1 0000E          ADDL3   #40, OBMODESC, -(SP)                ; 1072
                                  7E              5B  9A 00016          MOVZBL  COMMANDTYPE, -(SP)                  ; 1073
                                                  04  DD 00019          PUSHL   #4
                                      00000000G   8F  DD 0001B          PUSHL   #LIN$_ILLTIR
                       00000000G  00              06  FB 00021          CALLS   #6, LIB$SIGNAL
                                                  50  D4 00028          CLRL    R0                              ; 1075
                                                      05 0002A          RSB                                     ; 1076

; Routine Size:  43 bytes,    Routine Base:  $CODE$ + 01B0
```

```
;  502          1077  1 routine report_illtir2 : tir_proc2 =
;  503          1078  2 begin
;  504          1079  2
;  505          1080  2 external register
;  506          1081  2     commandtype = 11 : byte;
;  507          1082  2
;  508          1083  2 signal(lin$_illtir,4,
;  509          1084  2         .commandtype,obmodesc[omd$b_naming],
;  510          1085  2         .lnk$gl_record,lnk$gl_curfil[fdb$q_filename]);
;  511          1086  2
;  512          1087  2 return false
;  513          1088  1 end;


                                        0000 00000 REPORT_ILLTIR2:
                                                   .WORD   Save nothing                          ; 1077
                 7E 00000000G   00                 14   C1 00002   ADDL3   #20, LNK$GL_CURFIL, -(SP)   ; 1085
                                   00000000G       00   DD 0000A   PUSHL   LNK$GL_RECORD
                 7E 00000000'   EF                 28   C1 00010   ADDL3   #40, OBMODESC, -(SP)      ; 1084
                                7E                 5B   9A 00018   MOVZBL  COMMANDTYPE, -(SP)         ; 1085
                                                   04   DD 0001B   PUSHL   #4
                                   00000000G       8F   DD 0001D   PUSHL   #LIN$_ILLTIR
                    00000000G   00                 06   FB 00023   CALLS   #6, LIB$SIGNAL
                                                   50   D4 0002A   CLRL    R0                       ; 1087
                                                        04 0002C   RET                             ; 1088

; Routine Size:  45 bytes,    Routine Base:  $CODE$ + 01DB
```

```
;  515          1089  1 routine fndpscmapent(psctnum) =
;  516          1090  2 begin
;  517          1091  2 !       THIS ROUTINE RETURNS THE ADDRESS OF THE MAPPING TABLE ENTRY
;  518          1092  2 !       FOR THE GIVEN PSECT NUMBER.
;  519          1093  2 !
;  520          1094  2
;  521          1095  2 if .psctnum gtru .obmodesc[omd$w_hipsct]
;  522          1096  2     then return 0;
;  523          1097  2
;  524          1098  2 if not .obmodesc[omd$v_p256]
;  525          1099  2 then return obmodesc[omd$t_pscmap]+.psctnum*pmt$c_size
;  526          1100  2 !
;  527          1101  2 ! MODULE HAS MORE THAN 256 PSECTS
;  528          1102  2 !
;  529          1103  3 else begin
;  530          1104  3      local  mapent : ref block[,byte];
;  531          1105  3
;  532          1106  3       mapent = .(obmodesc[omd$t_pscmap] + (.psctnum/256)*pmt$c_size);    ! GET EXTENDED TABLE ADDRESS
;  533          1107  3       if .mapent neq 0                                                  ! IF EXTENDED TABLE PRESENT
;  534          1108  3       then return .mapent + (.psctnum mod 256)*pmt$c_size               !  THEN RETURN ENTRY ADDRESS
;  535          1109  3       else return 0;
;  536          1110  2       end;
;  537          1111  1 end;                                                                   ! OF FNDPSCMAP


                                0004 00000 FNDPSCMAPENT:
                                           .WORD   Save R2                                  ; 1089
                                52      04   AC  D0 00002          MOVL    PSCTNUM, R2     ; 1095
                                51 00000000' EF  D0 00006          MOVL    OBMODESC, R1    ;
      52      12   A1                 10      ED 00009          CMPZV   #0, #16, 18(R1), R2 ;
                                           2F  1F 00013          BLSSU   2$                  ;
              06       14   A1           06  E0 00015          BBS     #6, 20(R1), 1$      ; 1098
                                50    48 A142  7E 0001A          MOVAQ   72(R1)[R2], R0      ; 1099
                                           04  0001F          RET                         ; 1103
              50                52 00000100  8F  C7 00020 1$:   DIVL3   #256, R2, R0       ; 1106
                                   48 A140  7F  00028          PUSHAQ  72(R1)[R0]          ;
                                51         9E  D0 0002C          MOVL    @(SP)+, MAPENT      ;
                                           13  13 0002F          BEQL    2$                  ; 1107
      7E        00                52       01  7A 00031          EMUL    #1, R2, #0, -(SP)   ; 1108
      50        50                8E 00000100  8F  7B C0036          EDIV    #256, (SP)+, R0, R0 ;
                                50       6140  7E 0003F          MOVAQ   (MAPENT)[R0], R0    ;
                                           04  00043          RET                         ; 1109
                                      50   D4  00044 2$:        CLRL    R0                  ; 1111
                                           04  00046          RET                         ;

; Routine Size:  71 bytes,    Routine Base:  $CODE$ + 0208
```

```
;  539     1112  1 routine push (value, psectnum, posdeplvl) =
;  540     1113  2 begin
;  541     1114  2 !
;  542     1115  2 !       THIS ROUTINE IMPLEMENTS THE PUSHING OF VALUES ON THE LINKERS
;  543     1116  2 !       INTERNAL STACKS. INPUT PARAMETERS ARE:
;  544     1117  2 !               VALUE       =   VALUE TO BE PLACED ON STACK
;  545     1118  2 !               PSECTNUM    =   THE MODULE RELATIVE P-SECTION INDEX TO
;  546     1119  2 !                               WHICH THE STACKED VALUE IS RELATIVE
;  547     1120  2 !               POSDEPLVL   =   THE POSITION DEPENDENCE LEVEL OF THE VALUE
;  548     1121  2 !                               BEING STACKED.
;  549     1122  2 !                               NORMALLY = 1 FOR AN ADDRESS (E.G. A
;  550     1123  2 !                                            RELOCATED SYMBOL)
;  551     1124  2 !                                          0 FOR A SCALER.
;  552     1125  2 !       IF PSECTNUM = -1 THEN THE VALUE IS NOT TO BE RELATIVE TO A
;  553     1126  2 !       P-SECTION BASE ADDRESS. IF, HOWEVER, PSECTNUM IS NOT -1, THEN
;  554     1127  2 !       THE OBJECT MODULE DESCRIPTOR IS ACCESSED TO DETERMINE IF THIS
;  555     1128  2 !       MODULE HAS SUCH A P-SECTION (FOUND ON PASS 1) AND TO COMPUTE
;  556     1129  2 !       THE BASE ADDRESS OF THIS MODULE'S CONTRIBUTION TO THAT P-SECTION.
;  557     1130  2 !       IF THIS SUCCEEDS, THE VALUE STACKED IS THE SUM OF THE MODULE
;  558     1131  2 !       CONTRIBUTION BASE AND "VALUE"
;  559     1132  2 !
;  560     1133  2 !       IN ADDITION TO THE ABOVE CHECKS, THIS ROUTINE ALSO DETECTS STACK
;  561     1134  2 !       OVERFLOW. ON ANY FAILURE CONDITION, AN ERROR MESSAGE IS ISSUED
;  562     1135  2 !       AND THE VALUE "FALSE" IS RETURNED.
;  563     1136  2 !
;  564     1137  2 !       THERE ARE TWO PARALLEL STACKS - ONE FOR COMPUTATION AS DRIVEN
;  565     1138  2 !       BY THE COMPILER GENERATED OBJECT LANGUAGE AND THE OTHER
;  566     1139  2 !       CALLED THE PDL STACK, FOR LINKER COMPUTATION OF POSITION
;  567     1140  2 !       DEPENDENCE AND COERCION OF PIC IMAGES. THE COMPUTATION
;  568     1141  2 !       STACK IS LONGWORD ALIGNED WHILE THE PDL STACK IS A BYTE STACK.
;  569     1142  2 !
;  570     1143  2 local
;  571     1144  2         psctmap : ref block[,byte],             ! POINTER TO PSECT MAPPING TABLE ENTRY
;  572     1145  2         modcontribase,                          ! BASE OF THIS MODULES CONTRIBUTION TO P-SECTION
;  573     1146  2         psectdesc : ref block[,byte],           ! P-SECTION DESCRIPTOR POINTER
;  574     1147  2         modpscontriblk : ref block[,byte];      ! POINTER TO DESCRIPTOR OF CONTRIBUTION
;  575     1148  2 !
;  576     1149  2 if .psectnum eql -1                                             ! MODULE BASE OF P-SECTION
;  577     1150  3 then (modcontribase = 0; psectdesc = 0)                         ! IS ZERO IF NOT P-SECT RELATIVE
;  578     1151  3 else begin                                                      ! VALUE. IF IT IS PSECT
;  579     1152  3         if .psectnum gtru .obmodesc[omd$w_hipsct]                ! RELATIVE, CHECK P-SECTION
;  580     1153  3             then return report_udefpsc(.psectnum);
;  581     1154  3         psctmap = fndpscmapent(.psectnum);                      ! GET MAPPING TABLE ENTRY
;  582     1155  3         if .psctmap eql 0
;  583     1156  3             or (psectdesc = .psctmap[pmt$l_pscdes]) eql 0       ! GET P-SECTION DESCRIPTOR
;  584     1157  3             or (modpscontriblk = .psctmap[pmt$l_modcon]) eql 0  ! AND MODULE CONTRIBUTION BLOCK
;  585     1158  3             then return report_udefpsc(.psectnum);
;  586     1159  3         modcontribase = .psectdesc[psc$l_base] +                ! THE MODULE'S BASE IS
;  587     1160  3                                 .modpscontriblk[mpc$l_offset]; ! P-SECT BASE PLUS OFFSET.
;  588     1161  2     end;
;  589     1162  2 if .stackpointer eql 0                                                  ! NOW CHECK IF ALREADY
;  590     1163  3 then begin                                                             ! AT BOTTOM OF STACK AND
;  591     1164  3         signal(lin$_intstkov,3,stk$c_size,                             ! IF SO ISSUE ERROR MESSAGE
;  592     1165  3                         obmodesc[omd$b_namlng],
;  593     1166  3                         lnk$gl_curfil[fdb$q_filename]);
;  594     1167  3         return false;
;  595     1168  2         end;
```

```
;  596          1169  2
;  597          1170  2 stackpointer = .stackpointer -1;                          ! DECREMENT POINTER
;  598          1171  2 compustack[.stackpointer] = .value + .modcontribase;      ! PUSH VALUE ONTO COMPUTE STACK
;  599          1172  2 pdlstack[.stackpointer] = .posdeplvl;                     ! AND ON PDL STACK.
;  600          1173  2 psectstack[.stackpointer,0] = .psectdesc;                 ! SAVE PSECT DESCRIPTOR ADDRESS AND
;  601          1174  2 psectstack[.stackpointer,1] = .modcontribase;            ! MODULE'S BASE ON THE PSECT STACK
;  602          1175  3 shrsymstack[.stackpointer] = (if .shrimgsym neq 0        ! SAVE AWAY SHR IMG SYMBOL INFO
;  603          1176  3                               then .shrimgsym
;  604          1177  2                               else .shrsymexpr);
;  605          1178  2 stapxstack[.stackpointer] = .stapx;
;  606          1179  2 flagstack[.stackpointer] = .flags;
;  607          1180  2 return true;                                             ! AND RETURN SUCCESS.
;  608          1181  1 end;


                                                 007C 00000 PUSH:   .WORD   Save R2,R3,R4,R5,R6                  ; 1112
                                  56 00000000'   EF   9E 00002     MOVAB   STACKPOINTER, R6                      ;
                                  53       08    AC   D0 00009     MOVL    PSECTNUM, R3                          ; 1149
                             FFFFFFFF    8F         53   D1 0000D     CMPL    R3, #-1                               ;
                                                    04   12 00014     BNEQ    1$                                    ;
                                                    54   7C 00016     CLRQ    PSECTDESC                             ; 1150
                                                    2F   11 00018     BRB     4$                                    ;
                                  50       EF    A6   D0 0001A 1$:   MOVL    OBMODESC, R0                          ; 1152
          53       12   A0        10             00   ED 0001E     CMPZV   #0, #16, 18(R0), R3                   ;
                                                    15   1F 00024     BLSSU   2$                                    ;
                                                    53   DD 00026     PUSHL   R3                                    ; 1154
                        8D   AF                     01   FB 00028     CALLS   #1, FNDPSCMAPENT                      ;
                                                    50   D5 0002C     TSTL    PSCTMAP                               ; 1155
                                                    0B   13 0002E     BEQL    2$                                    ;
                                  54                60   D0 00030     MOVL    (PSCTMAP), PSECTDESC                  ; 1156
                                                    06   13 00033     BEQL    2$                                    ;
                                  52       04       A0   D0 00035     MOVL    4(PSCTMAP), MODPSCONTRIBLK            ; 1157
                                                    08   12 00039     BNEQ    3$                                    ;
                                                    53   DD 0003B 2$:   PUSHL   R3                                    ; 1158
                             FE10    CF             01   FB 0003D     CALLS   #1, REPORT_UDEFPSC                    ;
                                                    04   00042     RET                                           ;
          55       18   A4        08   A2   C1   00043 3$:   ADDL3   8(MODPSCONTRIBLK), 24(PSECTDESC), -         ; 1160
                                                                        MODCONTRIBASE                             ;
                                                    66   95 00049 4$:   TSTB    STACKPOINTER                          ; 1162
                                                    20   12 0004B     BNEQ    5$                                    ;
7E 00000000G       00                           14   C1 0004D     ADDL3   #20, LNK$GL_CURFIL, -(SP)             ; 1166
7E            EF   A6                           28   C1 00055     ADDL3   #40, OBMODESC, -(SP)                  ; 1165
                                                    19   DD 0005A     PUSHL   #25                                   ; 1166
                                                    03   DD 0005C     PUSHL   #3                                    ;
                                  00000000G         8F   DD 0005E     PUSHL   #LIN$_INTSTKOV                        ;
    00000000G     00                              05   FB 00064     CALLS   #5, LIB$SIGNAL                        ;
                                                    44   11 0006B     BRB     7$                                    ; 1167
                                                    66   97 0006D 5$:   DECB    STACKPOINTER                          ; 1170
                                  50                66   9A 0006F     MOVZBL  STACKPOINTER, R0                      ; 1171
                             13 A640          04 BC45   9E 00072     MOVAB   @VALUE[MODCONTRIBASE], COMPUSTACK[R0]  ;
                             77 A640          0C   AC   90 00079     MOVB    POSDEPLVL, PDLSTACK[R0]               ; 1172
                                        0093 C640       7F 0007F     PUSHAQ  PSECTSTACK[R0]                        ; 1173
                                  9E                54   D0 00084     MOVL    PSECTDESC, @(SP)+                     ;
                                        0097 C640       7F 00087     PUSHAQ  PSECTSTACK+4[R0]                      ; 1174
```

```
                9E              55  D0 0008C          MOVL    MODCONTRIBASE, @(SP)+
                51      E7      A6  D0 0008F          MOVL    SHRIMGSYM, R1                   ;1175
                                04  12 00093          BNEQ    6$
                51      EB      A6  D0 00095          MOVL    SHRSYMEXPR, R1                  ;1177
          015B C640             51  D0 00099 6$:      MOVL    R1, SHRSYMSTACK[R0]             ;1175
          01BF C640     FF      A6  90 0009F          MOVB    STAPX, STAPXSTACK[R0]           ;1178
          01DB C640     0F      A6  B0 000A6          MOVW    FLAGS, FLAGSTACK[R0]            ;1179
                50              01  D0 000AD          MOVL    #1, R0                          ;1180
                                    04 000B0          RET
                                50  D4 000B1 7$:      CLRL    R0                              ;1181
                                    04 000B3          RET

; Routine Size:  180 bytes,    Routine Base:  $CODE$ + 024F
```

```
;  610          1182  1 routine pop (valueaddr, pdladdr) =
;  611          1183  2 begin
;  612          1184  2 !
;  613          1185  2 !       THIS ROUTINE REMOVES VALUES FROM THE INTERNAL STACKS
;  614          1186  2 !       OF THE LINKER AS DESCRIBED WITH "PUSH" ABOVE.
;  615          1187  2 !       STACK OVERFLOW CAUSES THE ISSUANCE OF AN ERROR
;  616          1188  2 !       MESSAGE AND RETURN VALUE OF FALSE.
;  617          1189  2 !
;  618          1190  2 if .stackpointer gequ stk$c_size
;  619          1191  3 then begin
;  620          1192  3      signal(lin$_intstkun,3,stk$c_size,                      ! IF STACK IS ALREADY
;  621          1193  3                          obmodesc[omd$b_namlng],             ! EMPTY, ISSUE ERROR AND QUIT
;  622          1194  3                          lnk$gl_curfil[fdb$q_filename]);
;  623          1195  3       return false;
;  624          1196  2       end;
;  625          1197  2 .valueaddr    = .compustack [.stackpointer];                ! POP VALUE
;  626          1198  2 .pdladdr     = .pdlstack [.stackpointer];                  ! AND PDL TO WHERE CALLER WANTS THEM
;  627          1199  2 shrimgsym    = .shrsymstack [.stackpointer];               ! RESTORE SHR IMG SYMBOL INFO
;  628          1200  2 stapx        = .stapxstack [.stackpointer];
;  629          1201  2 flags        = .flagstack [.stackpointer];
;  630          1202  2 stackpointer = .stackpointer + 1;
;  631          1203  2 return true;                                               ! UPDATE POINTER AND
;  632          1204  1 end;                                                       ! SUCCESS

                                0004 00000 POP:     .WORD   Save R2                                 ; 1182
                    52 00000000'  FF  9E 00002      MOVAB   STACKPOINTER, R2                        ;
                    19            62  91 00009      CMPB    STACKPOINTER, #25                       ; 1190
                                  20  1F 0000C      BLSSU   1$                                      ;
                7E 00000000G      00  14 C1 0000E   ADDL3   #20, LNK$GL_CURFIL, -(SP)               ; 1194
                7E        EF      A2  28 C1 00016   ADDL3   #40, OBMODESC, -(SP)                    ; 1193
                                      19 DD 0001B   PUSHL   #25                                     ; 1194
                                      03 DD 0001D   PUSHL   #3                                      ;
                          00000000G   8F DD 0001F   PUSHL   #LIN$_INTSTKUN                          ;
                   00000000G      00  05 FB 00025   CALLS   #5, LIB$SIGNAL                          ;
                                      2A 11 0002C   BRB     2$                                      ; 1195
                          50          62 9A 0002E 1$: MOVZBL  STACKPOINTER, R0                      ; 1197
                    04    BC       13 A240  D0 00031   MOVL    COMPUSTACK[R0], @VALUEADDR          ;
                    08    BC       77 A240  9A 00037   MOVZBL  PDLSTACK[R0], @PDLADDR             ; 1198
                    E7    A2   015B C240  D0 0003D   MOVL    SHRSYMSTACK[R0], SHRIMGSYM          ; 1199
                    FF    A2   01BF C240  90 00044   MOVB    STAPXSTACK[R0], STAPX               ; 1200
                    0F    A2   01DB C240  B0 0004B   MOVW    FLAGSTACK[R0], FLAGS                ; 1201
                                      62 96 00052   INCB    STACKPOINTER                            ; 1202
                          50          01 D0 00054   MOVL    #1, R0                                  ; 1203
                                         04 00057   RET                                            ;
                                      50 D4 00058 2$: CLRL    R0                                    ; 1204
                                         04 0005A   RET                                             ;

; Routine Size:  91 bytes,    Routine Base:  $CODE$ + 0303
```

```
;  634          1205  1 routine compare_lit (value, node) =
;  635          1206  2 begin
;  636          1207  2 !
;  637          1208  2 ! COMPARE A VALUE WITH CURRENT NODE
;  638          1209  2 !
;  639          1210  2 map
;  640          1211  2     node : ref block[,byte];
;  641          1212  2
;  642          1213  3 return (.value - .node[lit$b_index])
;  643          1214  1 end;


                                        0000 00000 COMPARE_LIT:
                                                   .WORD   Save nothing                    ; 1205
                          50      08    AC  D0 00002       MOVL    NODE, R0               ; 1213
                          50      0B    A0  9A 00006       MOVZBL  11(R0), R0             ;
                    50        04    AC        50  C3 0000A SUBL3   R0, VALUE, R0          ;
                                              04 0000F     RET                            ; 1214

; Routine Size:  16 bytes,    Routine Base:  $CODE$ + 035E
```

```
;  645          1215  1 routine alloc_lit (value, retadr) =
;  646          1216  2 begin
;  647          1217  2 !
;  648          1218  2 ! ALLOCATE A NEW LIT BLOCK
;  649          1219  2 !
;  650          1220  2 map
;  651          1221  2     retadr : ref vector[,long];
;  652          1222  2
;  653          1223  2 lnk$alloblk(lit$c_size, retadr[0]);
;  654          1224  2
;  655          1225  2 return true
;  656          1226  1 end;


                                    0000 00000 ALLOC_LIT:
                                                       .WORD   Save nothing                    ; 1215
                                08    AC  DD 00002             PUSHL   RETADR                  ; 1223
                                      18  DD 00005             PUSHL   #24                     ;
                     00000000G  00        02  FB 00007        CALLS   #2, LNK$ALLOBLK         ;
                                50        01  D0 0000E        MOVL    #1, R0                  ; 1225
                                          04 00011           RET                             ; 1226

; Routine Size:  18 bytes,    Routine Base:  $CODE$ + 036E
```

```
;  658          1227  1 routine stackcommand : tir_proc =
;  659          1228  2 begin
;  660          1229  2 !
;  661          1230  2 ! PROCESS TIR STACK COMMANDS
;  662          1231  2 !
;  663          1232  2 external register
;  664          1233  2     commandtype = 11 : byte,
;  665          1234  2     commandata = 10 : ref block[,byte],
;  666          1235  2     datalng = 9;
;  667          1236  2
;  668          1237  2 local
;  669          1238  2     argnum,                                    ! INDEX TO ARGUMENT TO BE CHECKED
;  670          1239  2     argdesc : ref block[,byte],                ! POINTER TO ARGUMENT DESCRIPTOR
;  671          1240  2     psectnum,                                  ! P-SECTION NUMBER
;  672          1241  2     symdscnam : ref block[,byte],              ! PTR TO NAME BLOCK PART
;  673          1242  2     symdesc : ref block[,byte],                ! SYMBOL DESCRIPTOR POINTER
;  674          1243  2     envdesc : ref block[,byte],                ! POINTER TO ENVIRONMENT DESCRIPTOR
;  675          1244  2     value;                                     ! TEMP VALUE HOLDER
;  676          1245  2
;  677          1246  2 datalng = .stkdatlngtbl[.commandtype];                  ! GET EXPECTED DATA LENGTH
;  678          1247  2 if (commandata[signed_byte] + .datalng) gtru .endrecaddr! AND CHECK THAT IT EXISTS
;  679          1248  2     then return report_tirlng();
;  680          1249  2 psectnum = -1;                                         ! NOT P-SECTION RELATIVE
;  681          1250  2 shrimgsym = flags = pdl = value = stapx = 0;          ! AND NOT SHAREABLE IMAGE SYMBOL/PSECT
;  682          1251  2 !
;  683          1252  2 ! DISPATCH ON THE COMMAND TYPE
;  684          1253  2 !
;  685          1254  2 case .commandtype from 0 to tir$c_maxstacod
;  686          1255  2 of set
;  687          1256  2     [tir$c_sta_gbl,                                    ! STACK GLOBAL SYMBOL
;  688          1257  2     tir$c_sta_epm,                                     ! ENTRY POINT MASK
;  689          1258  2     tir$c_sta_ckarg] :                                 ! CHECK ARGUMENT
;  690          1259  3         begin
;  691          1260  3             datalng = .datalng+.commandata[symboleng];          ! UPDATE THE ACTUAL LENGTH OF THE DATA
;  692          1261  3             if .commandtype eql tir$c_sta_ckarg                 ! IF ARGUMENT CHECK COMMAND
;  693          1262  4             then begin
;  694          1263  5                 argnum = .(commandata[symboleng] + 1 +          ! FETCH THE ARGUMENT
;  695          1264  4                             .commandata[symboleng])<0,8,0>;     ! NUMBER AS AN UNSIGNED BYTE
;  696          1265  4                 argdesc = 2 + commandata[symboleng] +           ! AND COMPUTE ADDRESS
;  697          1266  4                                 .commandata[symboleng];         ! WITHIN RECORD
;  698          1267  4                 datalng = .datalng + .argdesc[arg$b_bytecnt];   ! THEN UPDATE DATA COUNT
;  699          1268  3                 end;
;  700          1269  3             if (commandata[symboleng] +.datalng) gtru .endrecaddr ! CHECK IF DATA ALL PRESENT
;  701          1270  3                 then return report_tirlng();
;  702          1271  3             if .commandata[symboleng] gtru sym$c_maxlng          ! CHECK SYMBOL NAME LENGTH
;  703          1272  3                 or .commandata[symboleng] eql 0
;  704          1273  3                 then return report_illnamlng(commandata[symboleng]);
;  705          1274  3             if not lnk$search(commandata[symboleng],symdesc,symdscnam) ! GO LOOK UP SYMBOL
;  706          1275  4             then begin
;  707          1276  4                 if .commandtype eql tir$c_sta_ckarg              ! IF NOT FOUND
;  708          1277  4                     then value = true;                           ! DEFAULT IS 0 EXCEPT FOR CHECK ARG
;  709          1278  4                 report_useundef(commandata[symboleng]);          ! REPORT REFERENCE TO UNDEFINDE SYMBOL
;  710          1279  4                 flags[flg$v_undef] = true;
;  711          1280  4                 end
;  712          1281  4             else begin                                           ! SYMBOL FOUND
;  713          1282  4                  bind imgclu = .symdesc[sym$l_cludsc] : block[,byte];
;  714          1283  4                 if .symdesc[sym$v_shrimg]                        ! IF SYMBOL IS FROM A SHAREABLE IMAGE
```

```
;  715          1284  4                    and .symdesc[sym$v_rel]                         !  AND IS RELOCATABLE
;  716          1285  4                    and not .symdesc[sym$v_gref]                    !  BUT HAS NOT BEEN PUT IN SHR LST
;  717          1286  4                    and not .imgclu[clu$v_based]
;  718          1287  4                    and not .lnk$gl_defclu[clu$v_based]
;  719          1288  5                    then begin
;  720          1289  5                        signal(lin$_nogsdref,3,commandata[symboleng], ! THEN TELL USER OF PROBLEM
;  721          1290  5                               obmodesc[omd$b_namlng],
;  722          1291  5                               lnk$gl_curfil[fdb$q_filename],lin$_noimgfil);
;  723          1292  5                        lnk$gl_ctlmsk[lnk$v_image] = false;
;  724          1293  4                        end;
;  725          1294  4                if .commandtype eql tir$c_sta_epm                   ! IF THIS IS STACK ENTRY POINT MASK
;  726          1295  4                    then if not .symdesc[sym$v_entmsk]              !  THEN MAKE SURE SYMBOL HAS ONE
;  727          1296  4                        then signal(lin$_noepm,3,                   !  ISSUE MESSAGE IF NOT.
;  728          1297  4                               commandata[symboleng],obmodesc[omd$b_namlng],
;  729          1298  4                               lnk$gl_curfil[fdb$q_filename]);      !  MASK FIELD IN SYMBOL BLOCK WILL BE 0
;  730          1299  4                if not .symdesc[sym$v_def]                          ! NOW CHECK WHAT WE FOUND WAS A DEF.
;  731          1300  4                    then
;  732          1301  5                        begin
;  733          1302  5                        flags[flg$v_undef] = true;
;  734          1303  5                        report_useundef(commandata[symboleng]);! AND REPORT ERROR IF NOT
;  735          1304  4                        end;
;  736          1305  4                if .commandtype eql tir$c_sta_gbl
;  737          1306  5                then begin
;  738          1307  5                    value = .symdesc[sym$l_value];                  ! GET SYMBOL VALUE IF COMMAND IS STACK GLOBA
;  739          1308  5                    if .symdesc[sym$v_rel]                          ! AND IF RELOCATABLE SET PDL = 1
;  740          1309  5                        then pdl = 1;
;  741          1310  5                    if .symdesc[sym$v_rel]                          ! IF SYMBOL IS RELOCATABLE
;  742          1311  5                        or .symdesc[sym$v_gref]                     ! OR FROM A SHARABLE IMAGE
;  743          1312  5                        then shrimgsym = .symdesc;                  ! REMEMBER ADDRESS OF SYMBOL BLOCK
;  744          1313  5                    end
;  745          1314  4                else if .commandtype eql tir$c_sta_epm          ! FOR STACK ENTRY POINT MASK, MASK IS THE VA
;  746          1315  4                    then value = .symdesc[sym$w_entmsk];
;  747          1316  4                if .commandtype eql tir$c_sta_ckarg
;  748          1317  5                then begin                                          ! CHECK ARGUMENT ...
;  749          1318  5                    bind
;  750          1319  5                        argvaldata = symdesc[sym$l_valdata]         ! POINT TO THE VALIDATION
;  751          1320  5                                   : ref vector[,byte];             ! DATA ARRAY
;  752          1321  5                    if .argvaldata eql 0                            ! BUT IF NONE
;  753          1322  5                        then value = true                           ! SET INDICATOR
;  754          1323  5                        else if (.argvaldata[0] - 3) lssu .argnum ! IF NOT ENOUGH ARGS
;  755          1324  5                            then value = true                       ! ALSO ASSUME MATCH
;  756          1325  5                            else if .argvaldata[2+.argnum]          ! OTHERWISE COMPARE THE ARGUMENTS
;  757          1326  5                                    eql .argdesc[arg$v_passmech]
;  758          1327  5                                then value = true;                  ! AND SET VALUE
;  759          1328  4                    end;
;  760          1329  3                end;
;  761          1330  2            end;
;  762          1331  2        [tir$c_sta_lsy,
;  763          1332  2         tir$c_sta_lepm] :
;  764          1333  3            begin
;  765          1334  3                datalng = .datalng + .commandata[lsymboleng];               ! COMPUTE SIZE OF DATA NEEDED
;  766          1335  3                if (commandata[lenvindex] + .datalng) gtru .endrecaddr      ! ENSURE ALL THERE
;  767          1336  3                    then return report_tirlng();                            !  AND RETURN WITH ERROR IF NOT
;  768          1337  3                if .commandata[lsymboleng] gtru sym$c_maxlng             ! CHECK LENGTH OF SYMBOL NAME
;  769          1338  3                    or .commandata[lsymboleng] eql 0
;  770          1339  3                    then return report_illnamlng(commandata[lsymboleng]);
;  771          1340  3                if not lnk$searchlocal(commandata[lsymboleng],          ! LOOK UP THE SYMBOL
```

```
;  772          1341  3                                .commandata[lenvindex],
;  773          1342  3                                symdesc,symdscnam,envdesc)
;  774          1343  4                        then begin
;  775          1344  4                            if .envdesc eql 0                                   ! SIGNAL UNDEFINED ENVIRONMENT IF TH
;  776          1345  4                                    or not .envdesc[nvd$v_def]
;  777          1346  4                                then report_useudfenv(.envdesc,.commandata[lenvindex]);
;  778          1347  4                            report_useudflsy(commandata[lsymboleng]);
;  779          1348  4                            end
;  780          1349  4                        else begin
;  781          1350  4                            if .envdesc eql 0
;  782          1351  4                                or not .envdesc[nvd$v_def]
;  783          1352  4                                then report_useudfenv(.envdesc,.commandata[lenvindex]);
;  784          1353  4                            if not .symdesc[sym$v_def]
;  785          1354  4                                then report_useudflsy(commandata[lsymboleng]);
;  786          1355  4                            if .commandtype eql tir$c_sta_lepm                  ! CHECK ENTRY MASK PRESENT IF STA_LE
;  787          1356  5                            then begin
;  788          1357  5                                if not .symdesc[sym$v_entmsk]
;  789          1358  5                                    then signal(lin$_noepm,3,commandata[lsymboleng],
;  790          1359  5                                            obmodesc[omd$b_namlng],lnk$gl_curfil[fdb$q_filename])
;  791          1360  5                                    else value = .symdesc[sym$w_entmsk];
;  792          1361  5                                end
;  793          1362  4                            else if .commandtype eql tir$c_sta_lsy              ! IF STACKING LOCAL SYMBOL, GET SYMB
;  794          1363  5                            then begin
;  795          1364  5                                value = .symdesc[sym$l_value];
;  796          1365  5                                if .symdesc[sym$v_rel]
;  797          1366  5                                    then pdl = 1;
;  798          1367  4                                end;
;  799          1368  3                            end;
;  800          1369  2                    end;
;  801          1370  2                [tir$c_sta_sb] :
;  802          1371  2                        value = .commandata[signed_byte];                       ! GET A SIGN EXTENDED BYTE
;  803          1372  2                [tir$c_sta_sw] :
;  804          1373  2                        value = .commandata[signed_word];                       ! GET SIGN EXTENDED WORD
;  805          1374  2                [tir$c_sta_lw] :
;  806          1375  2                        value = .commandata[longword];                          ! GET LONGWORD DATA
;  807          1376  2                [tir$c_sta_pb,
;  808          1377  2                tir$c_sta_pw,
;  809          1378  2                tir$c_sta_pl] :
;  810          1379  3                        begin
;  811          1380  3                        local
;  812          1381  3                            cludesc : ref block[,byte],
;  813          1382  3                            psctmap : ref block[,byte],
;  814          1383  3                            psctdesc : ref block[,byte];
;  815          1384  3
;  816          1385  3                        pdl = 1;                                                ! VALUE IS POSITION DEPENDENT
;  817          1386  3                        psectnum = .commandata[psect_index];                    ! EXTRACT PSECT NUMBER
;  818          1387  3                        psctmap = fndpscmapent(.psectnum);                      ! GET DESCRIPTOR ADDRESS                ;
;  819          1388  3                        if .psctmap neq 0
;  820          1389  4                        then begin
;  821          1390  4                            psctdesc = .psctmap[pmt$l_pscdes];                  ! GET PSECT DESCRIPTOR ADDRESS
;  822          1391  4                            shrimgsym = .psctdesc;                              ! SAVE PSECT DESCRIPTOR ADDRESS
;  823          1392  4                            cludesc = .psctdesc[psc$l_cludsc];                  ! POINT TO CLUSTER PSECT IS IN
;  824          1393  4                            if .cludesc[clu$v_shrimg]                           ! IF IT'S A SHAREABLE IMAGE CLUSTER
;  825          1394  4                                or .lnk$gl_ctlmsk[lnk$v_shr]                    !  OR CREATING SHR IMAGE
;  826          1395  4                                then stapx = true                               ! REMEBER LAST COMMAND WAS STAPX
;  827          1396  4                                else shrimgsym = 0;                             ! BUT IF NOT, JUST FORGET IT
;  828          1397  4                            if .cludesc[clu$v_shrimg]                           ! IF TARGET IS SHAREABLE IMAGE CLUSTER
```

```
;  829     1398  4                    and .cludesc[clu$v_based]                   ! WHICH IS BASED
;  830     1399  4                    then pdl = 0;                               ! THEN IT'S NOT POSITION DEPENDENT
;  831     1400  4                end
;  832     1401  3            else return report_udefpsc(.psectnum);              ! REPORT ERROR REFERENCING UNDEFINED PSECT
;  833     1402  3            case .commandtype from tir$c_sta_pb to tir$c_sta_pl
;  834     1403  3            of set
;  835     1404  3                [tir$c_sta_pb] :
;  836     1405  3                        value = .commandata[psect_b_offset];    ! GET A BYTE OFFSET
;  837     1406  3                [tir$c_sta_pw] :
;  838     1407  3                        value = .commandata[psect_w_offset];    ! WORD OFFSET
;  839     1408  3                [tir$c_sta_pl] :
;  840     1409  3                        value = .commandata[psect_l_offset];    ! LONGWORD OFFSET
;  841     1410  3                tes
;  842     1411  2            end;
;  843     1412  2        [tir$c_sta_wpb,                                         ! STACK PSECT BASE PLUS OFFSET COMMA
;  844     1413  2        tir$c_sta_wpw,                                          ! WHICH ALLOW A WORD OF PSECT NUMBER
;  845     1414  2        tir$c_sta_wpl] :
;  846     1415  3            begin
;  847     1416  3                local
;  848     1417  3                    cludesc : ref block[,byte],
;  849     1418  3                    psctmap : ref block[,byte],
;  850     1419  3                    psctdesc : ref block[,byte];
;  851     1420  3
;  852     1421  3                pdl = 1;                                        ! VALUE IS POSITION DEPENDENT
;  853     1422  3                psectnum = .commandata[wpsect_index];           ! EXTRACT PSECT NUMBER
;  854     1423  3                psctmap = fndpscmapent(.psectnum);              ! GET DESCRIPTOR ADDRESS
;  855     1424  3                if .psctmap neq 0
;  856     1425  4                then begin
;  857     1426  4                    psctdesc = .psctmap[pmt$l_pscdes];          ! GET PSECT DESCRIPTOR ADDRESS
;  858     1427  4                    shrimgsym = .psctdesc;                      ! SAVE PSECT DESCRIPTOR ADDRESS
;  859     1428  4                    cludesc = .psctdesc[psc$l_cludsc];          ! SAVE CLUSTER ADDRESS
;  860     1429  4                    if .cludesc[clu$v_shrimg]                   ! IF IT'S A SHAREABLE IMAGE CLUSTER
;  861     1430  4                        or .lnk$gl_ct[msk[lnk$v_shr]            !  OR CREATING SHR IMAGE
;  862     1431  4                        then stapx = true                       ! REMEBER LAST COMMAND WAS STAPX
;  863     1432  4                        else shrimgsym = 0;                     ! BUT IF NOT, JUST FORGET IT
;  864     1433  4                    if .cludesc[clu$v_shrimg]                   ! IF TARGET IS SHAREABLE IMAGE CLUST
;  865     1434  4                        and .cludesc[clu$v_based]               !  WHICH IS BASED
;  866     1435  4                        then pdl = 0;                           ! THEN IT'S NOT POSITION DEPENDENT
;  867     1436  4                end
;  868     1437  3                else return report_udefpsc(.psectnum);          ! NOT DEFINED, REPORT ERROR AND RETU
;  869     1438  3                case .commandtype from tir$c_sta_wpb to tir$c_sta_wpl
;  870     1439  3                of set
;  871     1440  3                    [tir$c_sta_wpb] :
;  872     1441  3                            value = .commandata[wpsect_b_offset];   ! GET A BYTE OFFSET
;  873     1442  3                    [tir$c_sta_wpw] :
;  874     1443  3                            value = .commandata[wpsect_w_offset];   ! WORD OFFSET
;  875     1444  3                    [tir$c_sta_wpl] :
;  876     1445  3                            value = .commandata[wpsect_l_offset];   ! LONGWORD OFFSET
;  877     1446  3                    tes
;  878     1447  2                end;
;  879     1448  2        [tir$c_sta_ub] :
;  880     1449  2                    value = .commandata[unsigned_byte];                 ! UNSIGNED BYTE
;  881     1450  2        [tir$c_sta_uw] :
;  882     1451  2                    value = .commandata[unsigned_word];                 ! UNSIGNED WORD
;  883     1452  2        [tir$c_sta_lit] :
;  884     1453  3                    begin
;  885     1454  3                        local
```

```
;  886          1455  3                    litindex;
;  887          1456
;  888          1457                  !
;  889          1458                  ! GET THE INDEX.  ENSURE THAT THE LITERAL
;  890          1459                  ! HAS BEEN DEFINED. USE 0 IF NOT, AND ISSUE
;  891          1460                  ! A WARNING MESSAGE
;  892          1461                  !
;  893          1462                  litindex = .commandata[unsigned_byte];
;  894          1463                  if not .lnk$gl_litflgs[.litindex]
;  895          1464                      then signal((lin$_stalitudf,4,.litindex,
;  896          1465                          .lnk$gl_record,obmodesc[omd$b_namlng],
;  897          1466                          lnk$gl_curfil[fdb$q_filename])
;  898          1467  4               else begin
;  899          1468  4                   local
;  900          1469  4                       entry : ref block[,byte];
;  901          1470  4
;  902          1471  4                   if lib$lookup_tree(lnk$gl_litbl,.litindex,          ! LOOKUP LITERAL IN TREE
;  903          1472  4                                      compare_lit,entry)
;  904          1473  5                   then begin
;  905          1474  5                       pdl = .entry[lit$v_pdl];                        ! RETRIEVE VALUES NEEDED
;  906          1475  5                       stapx = .entry[lit$v_stapx];
;  907          1476  5                       shrimgsym = .entry[lit$l_shrsym];
;  908          1477  5                       shrsymexpr = .entry[lit$l_shrexpr];
;  909          1478  5                       value = .entry[lit$l_value];
;  910          1479  5                       end
;  911          1480  4                   else signal(lin$_stalitudf,4,.litindex,
;  912          1481  4                               .lnk$gl_record,obmodesc[omd$b_namlng],
;  913          1482  4                               lnk$gl_curfil[fdb$q_filename]);
;  914          1483  3                   end;
;  915          1484  2               end;
;  916          1485  2           [inrange] :
;  917          1486  2                   return report_illtir2();                        ! OTHERWISE IT'S ILLEGAL
;  918          1487  2           tes;
;  919          1488  2
;  920          1489  2 return push (.value,.psectnum,.pdl)                               ! NOW PUSH THE VALUES AND RETURN
;  921          1490  1 end;


                                        1C  BB 00000 STACKCOMMAND:
                                                     PUSHR   #^M<R2,R3,R4>                  ; 1227
                                     5E             10  C2 00002          SUBL2   #16, SP
                                     50             5B  9A 00005          MOVZBL  COMMANDTYPE, R0                ; 1246
                                     59     FC73 CF40  9A 00008          MOVZBL  STKDATLNGTBL[R0], DATALNG
                                  50                59  C1 0000E          ADDL3   DATALNG, COMMANDATA, R0        ; 1247
                          00000000'  EF             50  D1 00012          CMPL    R0, ENDRECADDR
                                                    71  1A 00019          BGTRU   4$
                                     53             01  CE 0001B          MNEGL   #1, PSECTNUM                   ; 1249
                                        00000000'   EF  94 0001E          CLRB    STAPX                          ; 1250
                                                    58  D4 00024          CLRL    VALUE
                                        00000000'   EF  D4 00026          CLRL    PDL
                                        00000000'   EF  B4 0002C          CLRW    FLAGS
                                        00000000'   EF  D4 00032          CLRL    SHRIMGSYM
                                  13             00   8F 00038          CASEB   COMMANDTYPE, #0, #19           ; 1254
                          024C           0247          0242      0028     0003C 1$:      .WORD   2$-1$,-
```

```
                0334    0251        0251        0251    00044                   35$-1$,-                                                    ;
                03B4    03B4        03B4        0339    0004C                   36$-1$,-                                                    ;
                02BE    02BE        0028        0028    00054                   37$-1$,-                                                    ;
                0170    033E        0170        02BE    0005C                   39$-1$,-                                                    ;
                                                                                39$-1$,-                                                    ;
                                                                                39$-1$,-                                                    ;
                                                                                62$-1$,-                                                    ;
                                                                                64$-1$,-                                                    ;
                                                                                70$-1$,-                                                    ;
                                                                                70$-1$,-                                                    ;
                                                                                70$-1$,-                                                    ;
                                                                                2$-1$,-                                                     ;
                                                                                2$-1$,-                                                     ;
                                                                                50$-1$,-                                                    ;
                                                                                50$-1$,-                                                    ;
                                                                                50$-1$,-                                                    ;
                                                                                19$-1$,-                                                    ;
                                                                                66$-1$,-                                                    ;
                                                                                19$-1$                                                      ;
                                50              6A  9A 00064 2$:    MOVZBL  (COMMANDATA), R0                                            ; 1260
                                59              50  C0 00067        ADDL2   R0, DATALNG                                                 ;
                                0D              5B  91 0006A        CMPB    COMMANDTYPE, #13                                            ; 1261
                                                12  12 0006D        BNEQ    3$                                                          ;
                          55                    50  C1 0006F        ADDL3   R0, COMMANDATA, R5                                          ; 1263
                                54        01    A5  9A 00073        MOVZBL  1(R5), ARGNUM                                               ;
                                55              02  C0 00077        ADDL2   #2, ARGDESC                                                 ; 1265
                                50        01    A5  9A 0007A        MOVZBL  1(ARGDESC), R0                                              ; 1267
                                59              50  C0 0007E        ADDL2   R0, DATALNG                                                 ;
                          50                    59  C1 00081 3$:    ADDL3   DATALNG, COMMANDATA, R0                                     ; 1269
                00000000'   EF                  50  D1 00085        CMPL    R0, ENDRECADDR                                              ;
                                                03  1B 0008C 4$:    BLEQU   5$                                                          ;
                                              012F  31 0008E        BRW     20$                                                         ;
                                1F              6A  91 00091 5$:    CMPB    (COMMANDATA), #31                                           ; 1271
                                                04  1A 00094        BGTRU   6$                                                          ;
                                                6A  95 00096        TSTB    (COMMANDATA)                                                ; 1272
                                                05  12 00098        BNEQ    7$                                                          ;
                                                5A  DD 0009A 6$:    PUSHL   COMMANDATA                                                  ; 1273
                                              0137  31 0009C        BRW     23$                                                         ;
                                          04    AE  9F 0009F 7$:    PUSHAB  SYMDSCNAM                                                   ; 1274
                                          0C    AE  9F 000A2        PUSHAB  SYMDESC                                                     ;
                                                5A  DD 000A5        PUSHL   COMMANDATA                                                  ;
                00000000G       00              03  FB 000A7        CALLS   #3, LNK$SEARCH                                              ;
                                19              50  E8 000AE        BLBS    R0, 9$                                                      ;
                                0D              5B  91 000B1        CMPB    COMMANDTYPE, #13                                            ; 1276
                                                03  12 000B4        BNEQ    8$                                                          ;
                                58              01  D0 000B6        MOVL    #1, VALUE                                                   ; 1277
                                                5A  DD 000B9 8$:    PUSHL   COMMANDATA                                                  ; 1278
                      FC88      CF              01  FB 000BB        CALLS   #1, REPORT_USEUNDEF                                         ;
                00000000'       EF              01  88 000C0        BISB2   #1, FLAGS                                                   ; 1279
                                              032D  31 000C7        BRW     71$                                                         ; 1274
                                56        08    AE  D0 000CA 9$:    MOVL    SYMDESC, R6                                                 ; 1282
                                50        20    A6  D0 000CE        MOVL    32(R6), R0                                                  ;
                                57        0A    A6  9E 000D2        MOVAB   10(R6), R7                                                  ; 1283
          41                    67              0B  E1 000D6        BBC     #11, (R7), 10$                                              ;
          3D                    67              03  E1 000DA        BBC     #3, (R7), 10$                                               ; 1284
          39                    67              0E  E0 000DE        BBS     #14, (R7), 10$                                              ; 1285
                                35        58    A0  E8 000E2        BLBS    88(R0), 10$                                                 ; 1286
```

```
                 2E 00000000G   00   E8 000E6          BLBS    LNK$GL_DEFCLU+88, 10$                    ; 1287
                    00000000G   8F   DD 000ED          PUSHL   #LIN$_NOIMGFIL                           ; 1291
          7E 0C0000000G   00    14   C1 000F3          ADDL3   #20, LNK$GL_CURFIL, -(SP)
          7E 00000000'    EF    28   C1 000FB          ADDL3   #40, OBMODESC, -(SP)                     ; 1290
                                5A   DD 00103          PUSHL   COMMANDATA                               ; 1291
                                03   DD 00105          PUSHL   #3
                    00000000G   8F   DD 00107          PUSHL   #LIN$_NOGSDREF
             00000000G  00      06   FB 0010D          CALLS   #6, LIB$SIGNAL
             00000000G  00      01   8A 00114          BICB2   #1, LNK$GL_CTLMSK                        ; 1292
                                52   D4 0011B 10$:      CLRL    R2                                       ; 1294
                        0C      5B   91 0011D          CMPB    COMMANDTYPE, #12
                                27   12 00120          BNEQ    11$
                                52   D6 00122          INCL    R2
                                67   B5 00124          TSTW    (R7)                                     ; 1295
                                21   19 00126          BLSS    11$
          7E 00000000G  00      14   C1 00128          ADDL3   #20, LNK$GL_CURFIL, -(SP)                ; 1298
          7E 00000000'  EF      28   C1 00130          ADDL3   #40, OBMODESC, -(SP)                     ; 1297
                                5A   DD 00138          PUSHL   COMMANDATA                               ; 1298
                                03   DD 0013A          PUSHL   #3
                    00000000G   8F   DD 0013C          PUSHL   #LIN$_NOEPM
             00000000G  00      05   FB 00142          CALLS   #5, LIB$SIGNAL
          0E        67          01   E0 00149 11$:      BBS     #1, (R7), 12$                            ; 1299
             00000000'  EF      01   88 0014D          BISB2   #1, FLAGS                                ; 1302
                                5A   DD 00154          PUSHL   COMMANDATA                               ; 1303
                  FBED  CF      01   FB 00156          CALLS   #1, REPORT_USEUNDEF
                                5B   95 0015B 12$:      TSTB    COMMANDTYPE                              ; 1305
                                1F   12 0015D          BNEQ    15$
                        58      66   D0 0015F          MOVL    (R6), VALUE                              ; 1307
          0B            67      03   E1 00162          BBC     #3, (R7), 13$                            ; 1308
             00000000'  EF      01   D0 00166          MOVL    #1, PDL                                  ; 1309
          04            67      03   E0 0016D          BBS     #3, (R7), 14$                            ; 1310
          10            67      0E   E1 00171 13$:      BBC     #14, (R7), 16$                           ; 1311
             00000000'  EF      56   D0 00175 14$:      MOVL    R6, SHRIMGSYM                            ; 1312
                                07   11 0017C          BRB     16$                                      ; 1305
                        04      52   E9 0017E 15$:      BLBC    R2, 16$                                  ; 1314
                        58   08 A6   3C 00181          MOVZWL  8(R6), VALUE                             ; 1315
                        0D      5B   91 00185 16$:      CMPB    COMMANDTYPE, #13                         ; 1316
                                20   12 00188          BNEQ    18$
                        50   18 A6   D0 0018A          MOVL    24(R6), R0                               ; 1321
                                17   13 0018E          BEQL    17$
                        51      60   9A 00190          MOVZBL  (R0), R1                                 ; 1323
                        51      03   C2 00193          SUBL2   #3, R1
                        54      51   D1 00196          CMPL    R1, ARGNUM
                        0C      1F   00199          BLSSU   17$
                        51   02 A440 9A 0019B          MOVZBL  2(ARGNUM)[R0], R1                        ; 1326
          51        65    02      00   ED 001A0          CMPZV   #0, #2, (ARGDESC), R1
                                68   12 001A5          BNEQ    28$
                        58      01   D0 001A7 17$:      MOVL    #1, VALUE                                ; 1327
                                63   11 001AA 18$:      BRB     28$                                      ; 1316
                        50   02 AA   9A 001AC 19$:      MOVZBL  2(COMMANDATA), R0                        ; 1334
                        59      50   C0 001B0          ADDL2   R0, DATALNG
                     50         5A   C1 001B3          ADDL3   DATALNG, COMMANDATA, R0                  ; 1335
                00000000'       EF   D1 001B7          CMPL    R0, ENDRECADDR
                                07   1B 001BE          BLEQU   21$
                      FB30      CF   00   FB 001C0 20$:      CALLS   #0, REPORT_TIRLNG                    ; 1336
                                14   11 001C5          BRB     24$
                        52   02 AA   9E 001C7 21$:      MOVAB   2(COMMANDATA), R2                        ; 1337
```

```
                1F              62  91 001CB          CMPB    (R2), #31                       ;
                                04  1A 001CE          BGTRU   22$                             ;
                                62  95 001D0          TSTB    (R2)                            ; 1338
                                0A  12 001D2          BNEQ    25$                             ;
                                52  DD 001D4 22$:     PUSHL   R2                              ; 1339
             FAE7               CF  01 FB 001D6 23$:  CALLS   #1, REPORT_ILLNAMLNG            ;
                              0228  31 001DB 24$:     BRW     72$                             ;
                                5E  DD 001DE 25$:     PUSHL   SP                              ; 1340
                            08  AE  9F 001E0          PUSHAB  SYMDSCNAM                       ;
                            10  AE  9F 001E3          PUSHAB  SYMDESC                         ;
                        7E      6A  3C 001E6          MOVZWL  (COMMANDATA), -(SP)             ; 1341
                                52  DD 001E9          PUSHL   R2                              ; 1340
        0V0000000G      00      05  FB 001EB          CALLS   #5, LNK$SEARCHLOCAL             ;
                        1C      50  E8 001F2          BLBS    R0, 29$                         ;
                        50      6E  D0 001F5          MOVL    ENVDESC, R0                     ; 1344
                                04  13 001F8          BEQL    26$                             ;
                        0A  10  A0  E8 001FA          BLBS    16(R0), 27$                     ; 1345
                        7E      6A  3C 001FE 26$:     MOVZWL  (COMMANDATA), -(SP)             ; 1346
                                50  DD 00201          PUSHL   R0                              ;
             FB99       CF      02  FB 00203          CALLS   #2, REPORT_USEUDFENV            ;
                                52  DD 00208 27$:     PUSHL   R2                              ; 1347
             FBE6       CF      01  FB 0020A          CALLS   #1, REPORT_USEUDFLSY            ;
                                7A  11 0020F 28$:     BRB     38$                             ; 1340
                        50      6E  D0 00211 29$:     MOVL    ENVDESC, R0                     ; 1350
                                04  13 00214          BEQL    30$                             ;
                        0A  10  A0  E8 00216          BLBS    16(R0), 31$                     ; 1351
                        7E      6A  3C 0021A 30$:     MOVZWL  (COMMANDATA), -(SP)             ; 1352
                                50  DD 0021D          PUSHL   R0                              ;
             FB7D       CF      02  FB 0021F          CALLS   #2, REPORT_USEUDFENV            ;
                    55      08  AE  D0 00224 31$:     MOVL    SYMDESC, R5                     ; 1353
07          0A      A5          01  E0 00228          BBS     #1, 10(R5), 32$                 ;
                                52  DD 0022D          PUSHL   R2                              ; 1354
             FBC1       CF      01  FB 0022F          CALLS   #1, REPORT_USEUDFLSY            ;
                        13          5B  91 00234 32$:  CMPB    COMMANDTYPE, #19                ; 1355
                                2F  12 00237          BNEQ    34$                             ;
                            0B  A5  95 00239          TSTB    11(R5)                          ; 1357
                                23  19 0023C          BLSS    33$                             ;
7E  00000000G       00          14  C1 0023E          ADDL3   #20, LNK$GL_CURFIL, -(SP)       ; 1359
7E  00000000'       EF          28  C1 00246          ADDL3   #40, OBMODESC, -(SP)            ;
                                52  DD 0024E          PUSHL   R2                              ;
                                03  DD 00250          PUSHL   #3                              ;
                    00000000G       8F  DD 00252          PUSHL   #LIN$_NOEPM                 ;
    00000000G       00              05  FB 00258          CALLS   #5, LIB$SIGNAL              ;
                                2A  11 0025F          BRB     38$                             ;
                        58      08  A5  3C 00261 33$:     MOVZWL  8(R5), VALUE                ; 1360
                                0081  31 00265          BRW     46$                             ; 1355
                        11          5B  91 00268 34$:     CMPB    COMMANDTYPE, #17             ; 1362
                                7C  12 0026B          BNEQ    46$                             ;
                        58          65  D0 0026D          MOVL    (R5), VALUE                 ; 1364
16          0A      A5          03  E1 00270          BBC     #3, 10(R5), 38$                 ; 1365
    00000000'       EF          01  D0 00275          MOVL    #1, PDL                         ; 1366
                                79  11 0027C          BRB     49$                             ; 1254
                        58          6A  98 0027E 35$:     CVTBL   (COMMANDATA), VALUE         ; 1371
                                74  11 00281          BRB     49$                             ;
                        58          6A  32 00283 36$:     CVTWL   (COMMANDATA), VALUE         ; 1373
                                6F  11 00286          BRB     49$                             ;
                        58          6A  D0 00288 37$:     MOVL    (COMMANDATA), VALUE         ; 1375
```

```
                                    6A  11 0028B 38$:     BRB     49$                                     ;
                00000000'   EF          01  D0 0028D 39$:     MOVL    #1, PDL                                 ; 1385
                            53          6A  9A 00294          MOVZBL  (COMMANDATA), PSECTNUM                  ; 1386
                                        53  DD 00297          PUSHL   PSECTNUM                                ; 1387
                    FBEA    CF          01  FB 00299          CALLS   #1, FNDPSCMAPENT                        ;
                                        50  D5 0029E          TSTL    PSCTMAP                                 ; 1388
                                        6B  13 002A0          BEQL    51$                                     ;
                            50          60  D0 002A2          MOVL    (PSCTMAP), PSCTDESC                     ; 1390
                00000000'   EF          50  D0 002A5          MOVL    PSCTDESC, SHRIMGSYM                     ; 1391
                            50      24  A0  D0 002AC          MOVL    36(PSCTDESC), CLUDESC                   ; 1392
            08      58      A0          02  E0 002B0          BBS     #2, 88(CLUDESC), 40$                    ; 1393
            09 00000000G    00          02  E1 002B5          BBC     #2, LNK$GL_CTLMSK, 41$                  ; 1394
                00000000'   EF          01  90 002BD 40$:     MOVB    #1, STAPX                               ; 1395
                                        06  11 002C4          BRB     42$                                     ;
                        00000000'   EF  D4  002C6 41$:     CLRL    SHRIMGSYM                               ; 1396
            0A      58      A0          02  E1 002CC 42$:     BBC     #2, 88(CLUDESC), 43$                    ; 1397
                            06      58  A0  E9 002D1          BLBC    88(CLUDESC), 43$                        ; 1398
                        00000000'   EF  D4  002D5          CLRL    PDL                                     ; 1399
            02              04          5B  8F 002DB 43$:     CASEB   COMMANDTYPE, #4, #2                     ; 1402
          0014          000D            0006    002DF 44$:     .WORD   45$-44$,-
                                                                      47$-44$,-
                                                                      48$-44$
                            58      01  AA  98 002E5 45$:     CVTBL   1(COMMANDATA), VALUE                    ; 1405
                                    007C    31 002E9 46$:     BRW     60$                                     ;
                            58      01  AA  32 002EC 47$:     CVTWL   1(COMMANDATA), VALUE                    ; 1407
                                    0080    31 002F0          BRW     63$                                     ;
                            58      01  AA  D0 002F3 48$:     MOVL    1(COMMANDATA), VALUE                    ; 1409
                                    007E    31 002F7 49$:     BRW     65$                                     ; 1402
                00000000'   EF          01  D0 002FA 50$:     MOVL    #1, PDL                                 ; 1421
                            53          6A  3C 00301          MOVZWL  (COMMANDATA), PSECTNUM                  ; 1422
                                        53  DD 00304          PUSHL   PSECTNUM                                ; 1423
                    FB7D    CF          01  FB 00306          CALLS   #1, FNDPSCMAPENT                        ;
                                        50  D5 0030B          TSTL    PSCTMAP                                 ; 1424
                                        3B  13 0030D 51$:     BEQL    55$                                     ;
                            50          60  D0 0030F          MOVL    (PSCTMAP), PSCTDESC                     ; 1426
                00000000'   EF          50  D0 00312          MOVL    PSCTDESC, SHRIMGSYM                     ; 1427
                            50      24  A0  D0 00319          MOVL    36(PSCTDESC), CLUDESC                   ; 1428
            08      58      A0          02  E0 0031D          BBS     #2, 88(CLUDESC), 52$                    ; 1429
            09 00000000G    00          02  E1 00322          BBC     #2, LNK$GL_CTLMSK, 53$                  ; 1430
                00000000'   EF          01  90 0032A 52$:     MOVB    #1, STAPX                               ; 1431
                                        06  11 00331          BRB     54$                                     ;
                        00000000'   EF  D4  00333 53$:     CLRL    SHRIMGSYM                               ; 1432
            16      58      A0          02  E1 00339 54$:     BBC     #2, 88(CLUDESC), 56$                    ; 1433
                            12      58  A0  E9 0033E          BLBC    88(CLUDESC), 56$                        ; 1434
                        00000000'   EF  D4  00342          CLRL    PDL                                     ; 1435
                                        0A  11 00348          BRB     56$                                     ; 1424
                                        53  DD 0034A 55$:     PUSHL   PSECTNUM                                ; 1437
                    F9D0    CF          01  FB 0034C          CALLS   #1, REPORT_UDEFPSC                      ;
                                    00B2    31 00351          BRW     72$                                     ;
            02              0E          5B  8F 00354 56$:     CASEB   COMMANDTYPE, #14, #2                    ; 1438
          0012          000C            0006    00358 57$:     .WORD   58$-57$,-
                                                                      59$-57$,-
                                                                      61$-57$
                            58      02  AA  98 0035E 58$:     CVTBL   2(COMMANDATA), VALUE                    ; 1441
                                        61  11 00362          BRB     67$                                     ;
                            58      02  AA  32 00364 59$:     CVTWL   2(COMMANDATA), VALUE                    ; 1443
                                        5B  11 00368 60$:     BRB     67$                                     ;
```

```
                                58          02    AA  D0 0036A 61$:    MOVL    2(COMMANDATA), VALUE                    ; 1445
                                                  08  11 0036E          BRB     65$                                     ; 1438
                                58                6A  9A 00370 62$:    MOVZBL  (COMMANDATA), VALUE                     ; 1449
                                                  79  11 00373 63$:    BRB     69$
                                58                6A  3C 00375 64$:    MOVZWL  (COMMANDATA), VALUE                     ; 1451
                                                  74  11 00378 65$:    BRB     69$
                                55                6A  9A 0037A 66$:    MOVZBL  (COMMANDATA), LITINDEX                  ; 1462
                        42 00000000'  EF          55  E1 0037D          BBC     LITINDEX, LNK$GL_LITFLGS, 68$           ; 1463
                                            0C    AE  9F 00385          PUSHAB  ENTRY                                   ; 1471
                                          FC52    CF  9F 00388          PUSHAB  COMPARE_LIT
                                                  55  DD 0038C          PUSHL   LITINDEX
                                    00000000'     EF  9F 0038E          PUSHAB  LNK$GL_LITBL
                            00000000G   00        04  FB 00394          CALLS   #4, LIB$LOOKUP_TREE
                                        29        50  E9 0039B          BLBC    R0, 68$
                                        50    0C  AE  D0 0039E          MOVL    ENTRY, R0                               ; 1474
00000000'  EF       0A  A0              07        00  EF 003A2          EXTZV   #0, #7, 10(R0), PDL
           51       0A  A0              01        07  EF 003AC          EXTZV   #7, #1, 10(R0), R1                      ; 1475
                    00000000'  EF             51  90 003B2          MOVB    R1, STAPX
                    00000000'  EF       0C    A0  7D 003B9          MOVQ    12(R0), SHRIMGSYM                       ; 1476
                                        58    14  A0  D0 003C1          MOVL    20(R0), VALUE                           ; 1478
                                                  30  11 003C5 67$:    BRB     71$                                     ; 1471
                        7E 00000000G    00        14  C1 003C7 68$:    ADDL3   #20, LNK$GL_CURFIL, -(SP)               ; 1482
                        7E 00000000'    EF        28  C1 003CF          ADDL3   #40, OBMODESC, -(SP)                    ; 1481
                                    00000000G     00  DD 003D7          PUSHL   LNK$GL_RECORD                           ; 1482
                                                  55  DD 003DD          PUSHL   LITINDEX
                                                  04  DD 003DF          PUSHL   #4
                                    00000000G     8F  DD 003E1          PUSHL   #LIN$_STALITUDF
                            00000000G   00        06  FB 003E7          CALLS   #6, LIB$SIGNAL
                                                  07  11 003EE 69$:    BRB     71$                                     ; 1254
                                FA66    CF        00  FB 003F0 70$:    CALLS   #0, REPORT_ILLTIR2                      ; 1486
                                                  0F  11 003F5          BRB     72$
                                    00000000'     EF  DD 003F7 71$:    PUSHL   PDL                                     ; 1489
                                                  53  DD 003FD          PUSHL   PSECTNUM
                                                  58  DD 003FF          PUSHL   VALUE
                        FAC9    CF                03  FB 00401          CALLS   #3, PUSH
                                5E                10  C0 00406 72$:    ADDL2   #16, SP                                 ; 1490
                                                  1C  BA 00409          POPR    #^M<R2,R3,R4>
                                                      05 0040B          RSB

; Routine Size:  1036 bytes,    Routine Base:  $CODE$ + 0380
```

```
;  923          1491  1 routine baseallimages =
;  924          1492  2 begin
;  925          1493  2 !
;  926          1494  2 ! THIS ROUTINE ALLOCATES VIRTUAL ADDRESS SPACE FOR FIXUP SECTION AND
;  927          1495  2 ! BASES ALL PIC, NON-BASED SHAREABLE IMAGES
;  928          1496  2 !
;  929          1497  2 local
;  930          1498  2     fixhdr : ref block[,byte],
;  931          1499  2     fixsize,
;  932          1500  2     cluster : ref block[,byte],
;  933          1501  2     symsnb : ref block[,byte],
;  934          1502  2     symbol : ref block[,byte];
;  935          1503  2
;  936          1504  2 if not .lnk$gl_ctlmsk[lnk$v_lbased]              ! IF WE HAVEN'T DONE THIS YET
;  937          1505  3 then begin
;  938          1506  3     if .lnk$gl_fixisd eql 0                     ! IF NO FIXUP SECTION YET CREATED
;  939          1507  3         then lnk$crefixisd();
;  940          1508  3     fixhdr = lnk$gl_fixisd[isl$t_hdrisd];       ! POINT TO PART BOUND FOR HEADER
;  941          1509  3     fixhdr[isd$w_pagcnt] = lnk$getfixsize();    ! COMPUTE SIZE OF SECTION
;  942          1510  3     if not lnk$allovirmem(.fixhdr[isd$v_vpn]^9,
;  943          1511  3                           .fixhdr[isd$w_pagcnt])
;  944          1512  4     then begin
;  945          1513  4         signal(lin$_insvirmem,2,.fixhdr[isd$w_pagcnt],
;  946          1514  4                fixup_section,lin$_noimgfil);
;  947          1515  4         lnk$gl_ctlmsk[lnk$v_image] = false;
;  948          1516  3         end;
;  949          1517  3     lnk$gl_fixisd[isl$v_memalo] = true;         ! FLAG FIXUP SECTION ALLOCATED
;  950          1518  3 !
;  951          1519  3 ! NOW FIND AND BASE ALL PIC NON-BASED SHAREABLE IMAGES
;  952          1520  3 !
;  953          1521  3     cluster = lnk$gl_clulst;
;  954          1522  3     while (cluster = .cluster[clu$l_nxtclu]) neq 0
;  955          1523  3     do if .cluster[clu$v_shrimg]
;  956          1524  3         and not .cluster[clu$v_based]
;  957          1525  4     then begin
;  958          1526  4         lnk$cluvirmem(.cluster);                ! ALLOCATE VIRT MEMORY FOR THIS CLUSTER
;  959          1527  4         lnk$gw_shriscts = .lnk$gw_shriscts -    ! DECREMENT NUMBER OF PIC SHAREABLE IMAGE ISECTS
;  960          1528  4                           .cluster[clu$l_nisects];
;  961          1529  3         end;
;  962          1530  3     if not .basedmsgout
;  963          1531  4     then begin
;  964          1532  4         basedmsgout = true;
;  965          1533  4         signal(lin$_basduerrs);                 ! TELL USER HIS IMAGE JUST GOT NAILED DOWN
;  966          1534  3         end;
;  967          1535  3     lnk$gl_ctlmsk[lnk$v_lbased] = true;         ! FLAG WE HAVE BASED THE WORLD
;  968          1536  3     lnk$gl_ctlmsk[lnk$v_picimg] = false;        ! IMAGE IS NOT PIC ANY MORE
;  969          1537  2     end;
;  970          1538  2 return false
;  971          1539  1 end;                                           ! OF BASEALLIMAGES


                                007C 00000 BASEALLIMAGES:
                                           .WORD   Save R2,R3,R4,R5,R6                                     ; 1491
                    56 00000000'   EF  9E 00002    MOVAB   BASEDMSGOUT, R6                                 ;
```

```
                                55 00000000G   00   9E 00009          MOVAB   LIB$SIGNAL, R5                                    ;
                                54 00000000G   00   9E 00010          MOVAB   LNK$GL_FIXISD, R4                                 ;
                                53 00000000G   00   9E 00017          MOVAB   LNK$GL_CTLMSK, R3                                 ;
                      03    03  A3             04   E1 0001E          BBC     #4, LNK$GL_CTLMSK+3, 1$                           ; 1504
                                               0096 31 00023          BRW     7$                                                ;
                                                 64 D5 00026 1$:      TSTL    LNK$GL_FIXISD                                     ; 1506
                                                 07 12 00028          BNEQ    2$                                                ;
                   00000000G   00                00 FB 0002A          CALLS   #0, LNK$CREFIXISD                                 ; 1507
                52             64                18 C1 00031 2$:      ADDL3   #24, LNK$GL_FIXISD, FIXHDR                        ; 1508
                   00000000G   00                00 FB 00035          CALLS   #0, LNK$GETFIXSIZE                                ; 1509
                         02    A2                50 B0 0003C          MOVW    R0, 2(FIXHDR)                                     ;
                               7E          02    A2 3C 00040          MOVZWL  2(FIXHDR), -(SP)                                  ; 1511
             50       04    A2                15 00 EF 00044          EXTZV   #0, #21, 4(FIXHDR), R0                            ; 1510
                            7E                50 09 78 0004A          ASHL    #9, R0, -(SP)                                     ;
                   00000000G   00                02 FB 0004E          CALLS   #2, LNK$ALLOVIRMEM                                ;
                               1E                50 E8 00055          BLBS    R0, 3$                                            ;
                                   00000000G     8F DD 00058          PUSHL   #LIN$_NOIMGFIL                                    ; 1513
                                   00000000'     EF 9F 0005E          PUSHAB  FIXUP_SECTION                                     ;
                                               7E    02    A2 3C 00064          MOVZWL  2(FIXHDR), -(SP)                          ;
                                                 02 DD 00068          PUSHL   #2                                                ;
                                   00000000G     8F DD 0006A          PUSHL   #LIN$_INSVIRMEM                                   ;
                                               65                05 FB 00070          CALLS   #5, LIB$SIGNAL                    ;
                                               63                01 8A 00073          BICB2   #1, LNK$GL_CTLMSK                 ; 1515
                                               50                64 D0 00076 3$:      MOVL    LNK$GL_FIXISD, R0                 ; 1517
                                                     14          02 88 00079          BISB2   #2, 20(R0)                        ;
                                               52 00000000G      00 9E 0007D          MOVAB   LNK$GL_CLULST, CLUSTER            ; 1521
                                               52                62 D0 00084 4$:      MOVL    (CLUSTER), CLUSTER                ; 1522
                                                                 1C 13 00087          BEQL    5$                                ;
                                        F6           58    A2 02 E1 00089          BBC     #2, 88(CLUSTER), 4$               ; 1523
                                                     F2          58    A2 E8 0008E    BLBS    88(CLUSTER), 4$                   ; 1524
                                                                 52 DD 00092          PUSHL   CLUSTER                           ; 1526
                                           00000000G   00        01 FB 00094          CALLS   #1, LNK$CLUVIRMEM                 ;
                                           00000000G   00              48    A2 A2 0009B    SUBW2   72(CLUSTER), LNK$GW_SHRISCTS      ; 1528
                                                                 DF 11 000A3          BRB     4$                                ; 1523
                                                           0C                66 E8 000A5 5$:      BLBS    BASEDMSGOUT, 6$           ; 1530
                                                           66                01 90 000A8          MOVB    #1, BASEDMSGOUT           ; 1532
                                                                 000000vvv   8F DD 000AB          PUSHL   #LIN$_BASDUERRS           ; 1533
                                                           65                01 FB 000B1          CALLS   #1, LIB$SIGNAL            ;
                                                     03    A3                10 88 000B4 6$:      BISB2   #16, LNK$GL_CTLMSK+3      ; 1535
                                                     02    A3                02 8A 000B8          BICB2   #2, LNK$GL_CTLMSK+2       ; 1536
                                                                 50 D4 000BC 7$:      CLRL    R0                                ; 1538
                                                                 04 000BE             RET                                       ; 1539

; Routine Size:  191 bytes,    Routine Base:  $CODE$ + 078C
```

```
;  973          1540  1 routine checkshrsym (symbolblock) =
;  974          1541  2 begin
;  975          1542  2 !
;  976          1543  2 !       THIS ROUTINE CHECKS THAT THE SYMBOL POINTED TO BY SYMBOLBLOCK
;  977          1544  2 !       IS NOT FROM A SHAREABLE IMAGE.  IF IT IS, AN ERROR MESSAGE IS
;  978          1545  2 !       ISSUED, SINCE THE SYMBOL WAS NOT REFERENCED POSITION INDEPENDENTLY.
;  979          1546  2 !       THEN, AN ATTEMPT IS MADE TO ALLOCATE MEMORY FOR THE CLUSTER, SO THAT
;  980          1547  2 !       THIS REFERENCE CAN BE RESOLVED.
;  981          1548  2 !
;  982          1549  2 map
;  983          1550  2     symbolblock : ref block[,byte];
;  984          1551  2
;  985          1552  2 local
;  986          1553  2     symsnb : ref block[,byte];
;  987          1554  2
;  988          1555  2 if .symbolblock neq 0
;  989          1556  3 then begin
;  990          1557  3     if not .stapx
;  991          1558  3         and .symbolblock[sym$v_gref]
;  992          1559  4     then begin
;  993          1560  4         symsnb = .symbolblock - .symbolblock[sym$b_namlng] - snb$c_fxdlen;
;  994          1561  4         baseallimages();
;  995          1562  4         signal(lin$_shrsymref,5,symsnb[snb$b_namlng],
;  996          1563  4                 lnk$gl_curpsc[psc$b_namlng],(.lnk$gl_locn-.curpsectbase),
;  997          1564  4                 obmodesc[omd$b_namlng],lnk$gl_curfil[fdb$q_filename]);
;  998          1565  4         return false
;  999          1566  4         end
; 1000          1567  3     else if .stapx
; 1001          1568  3         and .symbolblock[psc$v_shrimg]
; 1002          1569  4     then begin
; 1003          1570  4         baseallimages();
; 1004          1571  4         signal(lin$_shrpscref,5,symbolblock[psc$b_namlng],
; 1005          1572  4                 lnk$gl_curpsc[psc$b_namlng],(.lnk$gl_locn-.curpsectbase),
; 1006          1573  4                 obmodesc[omd$b_namlng],lnk$gl_curfil[fdb$q_filename]);
; 1007          1574  4         return false
; 1008          1575  4         end
; 1009          1576  2     end;
; 1010          1577  2
; 1011          1578  2 return true
; 1012          1579  1 end;


                                    01FC 00000 CHECKSHRSYM:
                                                        .WORD   Save R2,R3,R4,R5,R6,R7,R8                       ; 1540
                                58 00000000'   EF  9E 00002     MOVAB   LNK$GL_CURPSC, R8                       ;
                                57 00000000G   00  9E 00009     MOVAB   LNK$GL_LOCN, R7                         ;
                                56 00000000G   00  9E 00010     MOVAB   LNK$GL_CURFIL, R6                       ;
                                55     FF26    CF  9E 00017     MOVAB   BASEAL[IMAGES, R5                       ;
                                54 00000000'   EF  9E 0001C     MOVAB   STAPX, R4                               ;
                                53       04    AC  D0 00023     MOVL    SYMBOLBLOCK, R3                         ; 1555
                                               66  13 00027     BEQL    4$                                      ;
                                35             64  E8 00029     BLBS    STAPX, 2$                               ; 1557
                          2D        0B     A3  06  E1 0002C     BBC     #6, 11(R3), 1$                          ; 1558
                                52       0F    A3  9A 00031     MOVZBL  15(R3), R2                              ; 1560
```

```
                52              53              52  C3 00035          SUBL3   R2, R3, R2                                     ;
                                52              05  C2 00039          SUBL2   #5, SYMSNB                                      ;
                                65              00  FB 0003C          CALLS   #0, BASEALLIMAGES                               ; 1561
             7E                 66              14  C1 0003F          ADDL3   #20, LNK$GL_CURFIL, -(SP)                       ; 1564
             7E         F0      A4              28  C1 00043          ADDL3   #40, OBMODESC, -(SP)                            ;
             7E                 67          FC  A4  C3 00048          SUBL3   CURPSECTBASE, LNK$GL_LOCN, -(SP)                ; 1563
             7E                 68              2D  C1 0004D          ADDL3   #45, LNK$GL_CURPSC, -(SP)                       ;
                                            04  A2  9F 00051          PUSHAB  4(SYMSNB)                                       ; 1562
                                                05  DD 00054          PUSHL   #5                                              ; 1564
                            00000000G           8F  DD 00056          PUSHL   #LIN$_SHRSYMREF                                 ;
                                                28  11 0005C          BRB     3$                                              ;
                            2E                  64  E9 0005E  1$:     BLBC    STAPX, 4$                                       ; 1567
             29         0B  A3                  05  E1 00061  2$:     BBC     #5, 11(R3), 4$                                  ; 1568
                            65                  00  FB 00066          CALLS   #0, BASEALLIMAGES                               ; 1570
             7E                 66              14  C1 00069          ADDL3   #20, LNK$GL_CURFIL, -(SP)                       ; 1573
             7E         F0      A4              28  C1 0006D          ADDL3   #40, OBMODESC, -(SP)                            ;
             7E                 67          FC  A4  C3 00072          SUBL3   CURPSECTBASE, LNK$GL_LOCN, -(SP)                ; 1572
             7E                 68              2D  C1 00077          ADDL3   #45, LNK$GL_CURPSC, -(SP)                       ;
                                            2D  A3  9F 0007B          PUSHAB  45(R3)                                          ; 1571
                                                05  DD 0007E          PUSHL   #5                                              ; 1573
                            00000000G           8F  DD 00080          PUSHL   #LIN$_SHRPSCREF                                 ;
                00000000G   00                  07  FB 00086  3$:     CALLS   #7, LIB$SIGNAL                                  ;
                                                04  11 0008D          BRB     5$                                              ; 1574
                            50                  01  D0 0008F  4$:     MOVL    #1, R0                                          ; 1578
                                                04  00092              RET                                                     ;
                                                50  D4 00093  5$:     CLRL    R0                                              ; 1579
                                                04  00095              RET                                                     ;

; Routine Size:  150 bytes,    Routine Base:  $CODE$ + 084B
```

```
;1014     1580  1 routine setaddressref (cludesc, isdesc, locn) =
;1015     1581  2 begin
;1016     1582  2 !
;1017     1583  2 ! THIS ROUTINE STORES INFO NECESSARY FOR .ADDRESS FIXUPS.  IT STORES THE
;1018     1584  2 ! LOCATION (GIVEN BY ARGUMENT LOCN) TO BE FIXED UP IN THE .ADDRESS FIXUP
;1019     1585  2 ! INFO FOR THE GIVEN CLUSTER.
;1020     1586  2 !
;1021     1587  2 ! INPUTS:
;1022     1588  2 !
;1023     1589  2 !       CLUDESC         ADDRESS OF CLUSTER DESCRIPTOR FOR TARGET OF REFERENCE
;1024     1590  2 !       ISDESC          ADDRESS OF ISECT DESCRIPTOR WHERE REF IS STORED
;1025     1591  2 !       LOCN            LOCATION OF REF
;1026     1592  2 !
;1027     1593  2 map
;1028     1594  2     cludesc : ref block[,byte],
;1029     1595  2     isdesc : ref block[,byte];
;1030     1596  2
;1031     1597  2 local
;1032     1598  2     adrblock : ref vector[,long];
;1033     1599  2
;1034     1600  2 bind
;1035     1601  2     lastblock = cludesc[clu$l_lstadrl] : ref vector[,long];
;1036     1602  2
;1037     1603  2 adrblock = .lastblock;                          ! POINT AT CURRENT BLOCK
;1038     1604  2 if .cludesc[clu$l_adrleft] eql 0                ! IF BLOCK IS EMPTY
;1039     1605  3 then begin
;1040     1606  3     lnk$alloblk(clu$c_adrblock*4,adrblock);     ! ALLOCATE ANOTHER BLOCK
;1041     1607  3     lastblock[0] = .adrblock;                   ! LINK INTO THE LIST
;1042     1608  3     cludesc[clu$l_lstadrl] = .adrblock;         ! MAKE IT THE NEW LAST BLOCK
;1043     1609  3     adrblock[0] = 0;                            ! MAKE IT THE END OF THE LIST
;1044     1610  3     cludesc[clu$l_adrleft] = clu$c_adrblock-1;  ! SET COUNT OF SLOTS LEFT IN BLOCK
;1045     1611  2     end;
;1046     1612  2 adrblock[clu$c_adrblock - .cludesc[clu$l_adrleft]] ! SET LOCATION IN BLOCK
;1047     1613  2                                 = .locn;
;1048     1614  2 cludesc[clu$l_adrleft] = .cludesc[clu$l_adrleft] -1; ! COUNT ONE LESS IN BLOCK
;1049     1615  2 if .cludesc[clu$l_adrcnt] eql 0                 ! IF THIS IS FIRST REF TO THIS CLUSTER
;1050     1616  2     then lnk$gl_shradr = .lnk$gl_shradr + 1;    !  THEN COUNT ANOTHER CLUSTER
;1051     1617  2 cludesc[clu$l_adrcnt] = .cludesc[clu$l_adrcnt] + 1; ! COUNT REF IN THIS CLUSTER
;1052     1618  2 lnk$gl_adrefs = .lnk$gl_adrefs + 1;             ! COUNT THE REFERENCE
;1053     1619  2 if .isdesc neq 0                                ! ENSURE THERE IS AN ISECT
;1054     1620  3 then begin
;1055     1621  3     bind
;1056     1622  3         curhdrisd = isdesc[isl$t_hdrisd]        ! PART FOR HEADER
;1057     1623  3                                 : block[,byte];
;1058     1624  3
;1059     1625  3     if not .curhdrisd[isd$v_crf]                ! SECTION MUST BE CRF
;1060     1626  4     then begin
;1061     1627  4         curhdrisd[isd$v_crf] = true;            ! MAKE IT CRF
;1062     1628  4         if not .curhdrisd[isd$v_wrt]
;1063     1629  5         then begin
;1064     1630  5             curhdrisd[isd$v_wrt] = true;        ! MAKE SECTION WRITEABLE
;1065     1631  5             lnk$gl_curisd[isl$b_newprt] = prt$c_ur; ! TELL IACT TO MAKE IT RO LATER
;1066     1632  5             lnk$gl_curisd[isl$v_reprot] = true; ! FLAG SECTION NEEDS REPROTECTING
;1067     1633  5             lnk$gl_rprotisd = .lnk$gl_rprotisd + 1; ! COUNT THE REPROTECTION
;1068     1634  4             end;
;1069     1635  3         end;
;1070     1636  2     end;
```

```
; 1071          1637  2
; 1072          1638  2 return true
; 1073          1639  1 end;


                                000C 00000 SETADDRESSREF:
                                           .WORD   Save R2,R3                          ; 1580
                      52    04    AC  D0 00002          MOVL    CLUDESC, R2            ; 1601
                            40    A2  DD 00006          PUSHL   64(R2)                 ; 1603
                      53    38    A2  9E 00009          MOVAB   56(R2), R3             ; 1604
                                  63  D5 0000D          TSTL    (R3)                   ;
                                  1D  12 0000F          BNEQ    1$                     ;
                                  5E  DD 00011          PUSHL   SP                     ; 1606
                            7E  0200  8F  3C 00013          MOVZWL  #512, -(SP)        ;
                00000000G   00        02  FB 00018          CALLS   #2, LNK$ALLOBLK    ;
                      40    B2        6E  D0 0001F          MOVL    ADRBLOCK, @64(R2)  ; 1607
                      40    A2        6E  D0 00023          MOVL    ADRBLOCK, 64(R2)   ; 1608
                                  00    BE  D4 00027          CLRL    @ADRBLOCK          ; 1609
                            63    7F    8F  9A 0002A          MOVZBL  #127, (R3)         ; 1610
                            50          63  D0 0002E 1$:      MOVL    (R3), R0           ; 1612
                            50          50  CE 00031          MNEGL   R0, R0             ;
                            50        00 BE40 DE 00034          MOVAL   @ADRBLOCK[R0], R0  ;
                      0200  C0        0C    AC  D0 00039          MOVL    LOCN, 512(R0)      ; 1613
                                        63  D7 0003F          DECL    (R3)               ; 1614
                                  34    A2  D5 00041          TSTL    52(R2)             ; 1615
                                        06  12 00044          BNEQ    2$                 ;
                           00000000'    EF  D6 00046          INCL    LNK$GL_SHRADR      ; 1616
                                  34    A2  D6 0004C 2$:      INCL    52(R2)             ; 1617
                           00000000'    EF  D6 0004F          INCL    LNK$GL_ADREFS      ; 1618
                                  08    AC  D5 00055          TSTL    ISDESC             ; 1619
                                        2C  13 00058          BEQL    3$                 ;
                      50        08    AC        18  C1 0005A          ADDL3   #24, ISDESC, R0    ; 1622
                      22        08    A0        01  E0 0005F          BBS     #1, 8(R0), 3$      ; 1625
                                08    A0        02  88 00064          BISB2   #2, 8(R0)          ; 1627
                      19        08    A0        03  E0 00068          BBS     #3, 8(R0), 3$      ; 1628
                                08    A0        08  88 0006D          BISB2   #8, 8(R0)          ; 1630
                                      50 00000000G  00  D0 00071          MOVL    LNK$GL_CURISD, R0  ; 1631
                                16    A0        0F  90 00078          MOVB    #15, 22(R0)        ;
                                14    A0        01  88 0007C          BISB2   #1, 20(R0)         ; 1632
                                      00000000G   00  D6 00080          INCL    LNK$GL_RPROTISD    ; 1633
                                      50        01  D0 00086 3$:      MOVL    #1, R0             ; 1638
                                                04  00089          RET                        ; 1639

; Routine Size:  138 bytes,    Routine Base:  $CODE$ + 08E1
```

```
;  1075          1640  1 routine compare_spcr (value,node) =
;  1076          1641  2 begin
;  1077          1642  2 !
;  1078          1643  2 ! COMPARE A VALUE WITH CURRENT NODE
;  1079          1644  2 !
;  1080          1645  2 map
;  1081          1646  2     node : ref block[,byte];
;  1082          1647  2
;  1083          1648  3 return (.value - .node[spcr$l_offset])
;  1084          1649  1 end;


                                        0000 00000 COMPARE_SPCR:
                                                   .WORD   Save nothing                          ; 1640
                          50      08    AC  D0 00002         MOVL    NODE, R0                    ; 1648
                50      04    AC      0A    A0  C3 00006         SUBL3   10(R0), VALUE, R0           ;
                                                04 0000C         RET                                 ; 1649

; Routine Size:  13 bytes,    Routine Base:  $CODE$ + 096B
```

```
;  1086          1650  1 routine alloc_spcr (value,retadr) =
;  1087          1651  2 begin
;  1088          1652  2 !
;  1089          1653  2 ! ALLOCATE A NEW SPCR BLOCK
;  1090          1654  2 !
;  1091          1655  2 map
;  1092          1656  2     retadr : ref vector[,long];
;  1093          1657  2
;  1094          1658  2 local
;  1095          1659  2     entry : ref block[,byte];
;  1096          1660  2
;  1097          1661  2 lnk$alloblk(spcr$c_size,entry);                ! ALLOCATE NEW BLOCK
;  1098          1662  2 entry[spcr$l_offset] = .value;                 ! SET OFFSET IN IMAGE
;  1099          1663  2 entry[spcr$l_fixadr] = .lnk$gl_lastgadr;        ! SET FIXUP VECTOR VA
;  1100          1664  2 lnk$gl_lastgadr = .lnk$gl_lastgadr + 4;        ! UPDATE VA FOR NEXT
;  1101          1665  2 lnk$gl_spcgfix = .lnk$gl_spcgfix + 1;         ! COUNT ANOTHER SPECIAL FIXUP
;  1102          1666  2 retadr[0] = .entry;                           ! RETURN VALUE TO CALLER
;  1103          1667  2
;  1104          1668  2 return true
;  1105          1669  1 end;

                                0004 00000 ALLOC_SPCR:
                                           .WORD   Save R2                                          ; 1650
                    52 00000000G  00  9E 00002     MOVAB   LNK$GL_LASTGADR, R2                      ;
                    5E            04  C2 00009     SUBL2   #4, SP                                   ;
                                  5E  DD 0000C     PUSHL   SP                                       ; 1661
                                  12  DD 0000E     PUSHL   #18                                      ;
                00000000G         00  FB 00010     CALLS   #2, LNK$ALLOBLK                          ;
                                  50  6E D0 00017   MOVL    ENTRY, R0                                ; 1662
                      0A          A0      04  AC D0 0001A   MOVL    VALUE, 10(R0)                    ;
                      0E          A0          62 D0 0001F   MOVL    LNK$GL_LASTGADR, 14(R0)          ; 1663
                                  62          04 C0 00023   ADDL2   #4, LNK$GL_LASTGADR              ; 1664
                                      00000000' EF D6 00026 INCL    LNK$GL_SPCGFIX                   ; 1665
                      08          BC          50 D0 0002C   MOVL    R0, @RETADR                      ; 1666
                                  50          01 D0 00030   MOVL    #1, R0                           ; 1668
                                              04 00033      RET                                      ; 1669

; Routine Size:  52 bytes,    Routine Base:  $CODE$ + 0978
```

```
;  1107          1670  1 routine storecommand : tir_proc =
;  1108          1671  2 begin
;  1109          1672  2 external register
;  1110          1673  2     commandtype = 11 : byte,
;  1111          1674  2     commandata  = 10 : ref block [,byte],
;  1112          1675  2     datalng     = 9 ;
;  1113          1676  2
;  1114          1677  2 local
;  1115          1678  2     stocmdindex,                                    ! INDEX INTO THE GROUP
;  1116          1679  2     correctvalue,                                   ! CORRECTED STORE VALUE
;  1117          1680  2     outputaddr,                                     ! ADDRESS OF OUTPUT STREAM
;  1118          1681  2     mbzmask,                                        ! MASK COMPUTED TO CORRECT THE VALUE
;  1119          1682  2     outbytecount,                                   ! NUMBER OF BYTES TO WRITE OUT
;  1120          1683  2     lowestmbzbit,                                   ! FOR COMPUTING MBZ FIELDS
;  1121          1684  2     repeatcount,                                    ! STORE REPEATED COUNTS
;  1122          1685  2     outputarray : vector [5, byte] ;                ! OUTPUT DATA
;  1123          1686  2
;  1124          1687  2 bind
;  1125          1688  2     value = outputarray [1] ;                       ! LEAVE ONE BYTE FOR ADDRESSING MODE
;  1126          1689  2
;  1127          1690  2 stocmdindex  = .commandtype - tir$c_minstocod ;          ! CREATE USEFUL INDEX
;  1128          1691  2 outbytecount = .storectltbl [.stocmdindex,stoctl$b_bytes] ;   ! GET OUTPUT BYTE COUNT
;  1129          1692  2
;  1130          1693  3 if (.storectltbl [.stocmdindex,stoctl$b_flags]            ^   ! IF THIS IS A
;  1131          1694  3     and
;  1132          1695  3     stoctl$m_rep
;  1133          1696  2    ) neq 0                                  ! REPEATED STORE,
;  1134          1697  3 then begin
;  1135          1698  3      if not pop (repeatcount,pdl)                       ! GET TOP OF STACK
;  1136          1699  3      then return false ;                                 ! GIVING UP ON ERROR
;  1137          1700  3      end
;  1138          1701  2 else repeatcount = 1 ;                                  ! NO REPEAT, SO SET TO 1.
;  1139          1702  2
;  1140          1703  2 if  .commandtype eql tir$c_sto_rivb                     ! IF THIS IS STOREPEATED IMMEDIATE VAR. NO.
;  1141          1704  3 then begin                                             ! PROCESS IT NOW
;  1142          1705  3      bind  databytes = tirrec [.tiroffset+1] : vector [,byte] ;  ! NAME THE DATA PART
;  1143          1706  3
;  1144          1707  3      outbytecount = .databytes [0] ;                    ! GET THE OUTPUT BYTE COUNT
;  1145          1708  3      datalng      = .outbytecount + 1 ;                 ! FIGURE LENGTH OF RECORD NEEDED TO BE THERE
;  1146          1709  3      shrimgsym    = stapx = shrsymexpr = 0 ;            !
;  1147          1710  3
;  1148          1711  4      return (if (.outbytecount+databytes [1]) gtru .endrecaddr  ! IF RECORD IS TOO SHORT
;  1149          1712  4              then report_tirlng  ()
;  1150          1713  4              else lnk$wrtimgbyts (.outbytecount,databytes [1]   ! OTHERWISE, RETURN, WRITING THE BYTES TO TH
;  1151          1714  4                                  ,.repeatcount                  ! THE NUMBER OF TIMES SPECIFIED
;  1152          1715  4                                  )
;  1153          1716  3              ) ;
;  1154          1717  2      end ;
;  1155          1718  2
;  1156          1719  2 if not pop (value,pdl)                                 ! ATTEMPT TO GET
;  1157          1720  2 then return false ;                                    ! VALUE TO BE STORED
;  1158          1721  2
;  1159          1722  3 if (.pdl neq 0                                         ! IF POSITION-DEPENDENT
;  1160          1723  3     or
;  1161          1724  4     (not .stapx and (.shrimgsym neq 0) and .shrimgsym [sym$v_gref])
;  1162          1725  3    )
;  1163          1726  2    and .commandtype neq tir$c_sto_picr                  !  AND COMMAND IS NOT STORE PIC CODE
```

```
;  1164          1727  2     and .commandtype neq tir$c_sto_pidr                                  !  OR STORE PIC DATA REFERERNCE
;  1165          1728  2     and not .lnk$gl_ctlmsk [lnk$v_ubased]                                !  AND IMAGE NOT BASED BY USER
;  1166          1729  3 then begin
;  1167          1730  3       if not checkshrsym (.shrimgsym)                                    ! THEN CHECK CORRECTNESS OF REFERENCE
;  1168          1731  4       then value = (if not .stapx
;  1169          1732  5                     then (.shrimgsym [sym$l_value] + .absexpr)           ! IF RELOCATED, USE NEW VALUE
;  1170          1733  5                                                                          !  BUT DON'T FORGET THE REST OF THE EXPRN
;  1171          1734  4                     else 0                                               !**??**
;  1172          1735  4                    )
;  1173          1736  3       else if  .pdl eql 1                                                ! IF POS DEP LEVEL IS 1
;  1174          1737  3            and      .lnk$gl_ctlmsk [lnk$v_shr]                          ! AND CREATING SHR IMAGE
;  1175          1738  3            and not .lnk$gl_ctlmsk [lnk$v_ubased]
;  1176          1739  3            and  .commandtype neq tir$c_sto_pirr
;  1177          1740  4            and (.commandtype lssu tir$c_sto_bd                          ! AND COMMAND IS NOT STORE DISPLACED
;  1178          1741  4                 or
;  1179          1742  4                .commandtype gtru tir$c_sto_ld                          !  OF BYTE, WORD, OR LONGWORD
;  1180          1743  4                )
;  1181          1744  4            then  begin
;  1182          1745  4                  baseallimages () ;                                    ! THEN BASE EVERYTHING
;  1183          1746  4                  signal  (lin$_nonpicref,4,lnk$gl_curpsc [psc$b_namlng]  ! THEN THAT IS AN ERROR
;  1184          1747  5                          ,(.lnk$gl_locn-.curpsect$base)
;  1185          1748  4                          ,obmodesc [omd$b_namlng]
;  1186          1749  4                          ,lnk$gl_curfil [fdb$q_filename]
;  1187          1750  4                          ) ;
;  1188          1751  3                  end ;
;  1189          1752  3       stapx = shrimgsym = shrsymexpr = absexpr = 0 ;                     ! CLEAR OUT SHR IMG VARIABLES
;  1190          1753  2       end ;
;  1191          1754  2
;  1192          1755  2 if (.storectltbl [.stocmdindex,stoctl$b_flags] and stoctl$m_displ) neq 0        ! IF A DISPLACEMENT COMMAND
;  1193          1756  3 then begin
;  1194          1757  3       if  .pdl eql 0
;  1195          1758  3       and .lnk$gl_ctlmsk [lnk$v_shr]                                     ! then can't have non-relocatable
;  1196          1759  3       and .lnk$gl_ctlmsk [lnk$v_picimg]                                  !  references stored in pic images
;  1197          1760  3       and not .flags [flg$v_undef]                                       ! (special case undefined syms - act like no
;  1198          1761  3       then signal (lin$_stdispic, 4, lnk$gl_curpsc [psc$b_namlng]
;  1199          1762  4                  ,(.lnk$gl_locn - .curpsect$base)
;  1200          1763  3                  ,obmodesc [omd$b_namlng]
;  1201          1764  3                  ,lnk$gl_curfil [fdb$q_filename]
;  1202          1765  3                  ) ;
;  1203          1766  3                                                                          ! subtract current location counter and
;  1204          1767  3       value = .value - .lnk$gl_locn - .outbytecount ;                    !   output byte count
;  1205          1768  2       end ;
;  1206          1769  2
;  1207          1770  2 lowestmbzbit = .storectltbl [.stocmdindex,stoctl$v_mbzbit] ;            ! GET NUMBER OF LOWEST
;  1208          1771  2
;  1209          1772  3 if (.storectltbl [.stocmdindex,stoctl$b_flags]
;  1210          1773  3       and
;  1211          1774  3      stoctl$m_conmbz
;  1212          1775  2    ) neq 0                                                               ! BIT WHICH MUST BE ZERO
;  1213          1776  2 then if .value <.lowestmbzbit,1,0> neq 0                                ! IF A CONDITIONALLY MBZ
;  1214          1777  2       then lowestmbzbit = 0 ;                                           ! FIELD, SUPPRESS MBZ IF VALUE IS NEGATIVE
;  1215          1778  2
;  1216          1779  2 if  .lowestmbzbit eql 0                                                ! IF THERE IS NOT AN MBZ FIELD
;  1217          1780  2 then correctvalue = .value < 0,min (32,8*.outbytecount),1>             ! CORRECT VALUE IS SIGN EXTENSION
;  1218          1781  3 else begin                                                             ! OF THE FIELD TO BE WRITTEN
;  1219          1782  3       mbzmask = (1^.lowestmbzbit) -1 ;                                  ! OTHERWISE, MASK VALUE TO GET CORRECT
;  1220          1783  3       correctvalue = .value and .mbzmask ;                              ! ONE
```

```
;  1221          1784  2      end ;
;  1222          1785  2
;  1223          1786  2 if  .value neq .correctvalue                          . IF VALUE DOES NOT MATCH
;  1224          1787  3 then begin
;  1225          1788  3      signal (lin$_trunc, 4                            ! NOT MATCH THE CORRECT VERSION
;  1226          1789  3             ,lnk$gl_curpsc [psc$b_namlng]
;  1227          1790  4             ,(.lnk$gl_locn - .curpsectbase)           ! ISSUE TRUNCATION ERROR
;  1228          1791  3             ,obmodesc [omd$b_namlng]
;  1229          1792  3             ,lnk$gl_curfil [fdb$q_filename]
;  1230          1793  3             ,lin$_truncdat, 3                         ! GIVING MODULE, PSECTION,
;  1231          1794  3             ,.value, .correctvalue                    ! OFFSET, THEN COMPUTED VALUE,
;  1232          1795  3             ,.lnk$gl_locn                             ! WRITTEN VALUE AND VIRTUAL ADDRESS
;  1233          1796  3             ) ;
;  1234          1797  3      value = .correctvalue ;                          ! GET THE CORRECT VALUE
;  1235          1798  2      end ;
;  1236          1799  2
;  1237          1800  2 if .commandtype eql tir$c_sto_vps                     ! STORE VARIABLE POSITION AND SIZE
;  1238          1801  3 then begin
;  1239          1802  3      local       vpspos,
;  1240          1803  3                  vpssiz ;
;  1241          1804  3
;  1242          1805  3      datalng = 2 ;
;  1243          1806  3      if tirrec [.tiroffset+2] gtru .endrecaddr        ! MUST HAVE TWO MORE BYTES FOR COMMAND
;  1244          1807  3                                                       !% missing dot?
;  1245          1808  3      then return report_tirlng () ;
;  1246          1809  3
;  1247          1810  3      vpspos = .tirrec [.tiroffset+1] ;                ! GET POSITION
;  1248          1811  3      vpssiz = .tirrec [.tiroffset+2] ;                ! AND SIZE
;  1249          1812  3
;  1250          1813  3      if .vpspos+.vpssiz gequ 32                       ! CHECK LEGAL
;  1251          1814  4      or .vpspos+.vpssiz gtru ((.vpspos+8)/8*8)
;  1252          1815  4      then begin
;  1253          1816  4           signal (lin$_illvps, 4, .vpspos, .vpssiz
;  1254          1817  4                  ,obmodesc [omd$b_namlng]
;  1255          1818  4                  ,lnk$gl_curfil [fdb$q_filename]
;  1256          1819  4                  ) ;
;  1257          1820  4            return false
;  1258          1821  4            end
;  1259          1822  3      else return lnk$wrtimgbyts (1,.value,1,.vpspos,.vpssiz) ;
;  1260          1823  2      end ;
;  1261          1824  2
;  1262          1825  2 if  .commandtype eql tir$c_sto_pirr                    ! STORE POSITION INDEPENDENT RELATIVE REFERE
;  1263          1826  3 then begin
;  1264          1827  3      local       value2,
;  1265          1828  3                  pdl2 ;
;  1266          1829  3
;  1267          1830  3      if not pop (value2, pdl2)                        ! POP THE RELATIVE LOCATION
;  1268          1831  3      then return false ;                               ! BUT IF AN ERROR, GIVE UP
;  1269          1832  3
;  1270          1833  4      if (.pdl2 neq 0                                  ! IF POS DEP VALUE
;  1271          1834  4          or
;  1272          1835  5          (not .stapx and (.shrimgsym neq 0) and .shrimgsym [sym$v_gref])
;  1273          1836  4          )
;  1274          1837  3          and not .lnk$gl_ctlmsk [lnk$v_ubased]
;  1275          1838  4      then begin
;  1276          1839  4           if not checkshrsym (.shrimgsym)
;  1277          1840  5           then value2 = (if not .stapx
```

```
;  1278     1841  5                        then .shrimgsym [sym$l_value]
;  1279     1842  5                        else 0
;  1280     1843  5                       )
;  1281     1844  4            else  if  .pdl2 eql 1
;  1282     1845  4                  and     .lnk$gl_ctlmsk [lnk$v_shr]
;  1283     1846  4                  and not .lnk$gl_ctlmsk [lnk$v_ubased]
;  1284     1847  5                  then begin
;  1285     1848  5                       baseallimages () ;                          ! BASE THE WORLD
;  1286     1849  5                       signal (lin$_nonpicref, 4
;  1287     1850  5                              ,lnk$gl_curpsc [psc$b_namlng]
;  1288     1851  6                              ,(.lnk$gl_locn-.curpsectbase)
;  1289     1852  5                              ,obmodesc [omd$b_namlng]
;  1290     1853  5                              ,lnk$gl_curfil [fdb$q_filename]
;  1291     1854  5                              ) ;
;  1292     1855  4                       end ;
;  1293     1856  3             end ;
;  1294     1857  3
;  1295     1858  3       if .value2 eql -1 then value2 = .lnk$gl_locn ;            ! SUBSTITUTUE CURRENT LOC. FOR -1
;  1296     1859  3       if .pdl neq 0     then value  = .value - .value2 ;        ! RELOCATE VALUE IF CALLED FOR.
;  1297     1860  2       end ;
;  1298     1861  2
;  1299     1862  2 if  .commandtype neq tir$c_sto_picr                            ! IF THIS IS NOT A STORE POS. IND. CODE OR D
;  1300     1863  2 and .commandtype neq tir$c_sto_pidr
;  1301     1864  2 then outputaddr = outputarray [1]                             ! THEN SET THE OUTPUT POINTER
;  1302     1865  2 else if .commandtype eql tir$c_sto_pidr                       ! IF A STORE PIC DATA REF
;  1303     1866  3       then begin
;  1304     1867  4            bind  symclu = (if .stapx                          ! NAME THE CLUSTER THAT REF. IS IN
;  1305     1868  4                            then shrimgsym [psc$l_cludsc]
;  1306     1869  4                            else shrimgsym [sym$l_cludsc]
;  1307     1870  3                            ) . ref block [,byte] ;
;  1308     1871  3
;  1309     1872  3            outputaddr = outputarray [1] ;                     ! FOUR BYTES GO
;  1310     1873  3            if .pdl neq 0                                      ! IF NON-PIC DATA REFERENCE
;  1311     1874  3            and not .lnk$gl_ctlmsk [lnk$v_dbgrec]              !  AND NOT IN A DEBUG RECORD
;  1312     1875  4            then  begin
;  1313     1876  5                  if begin
;  1314     1877  5                     if  .lnk$gl_ctlmsk [lnk$v_shr]          ! OTHERWISE IF CREATING A SHAREABLE IMAGE
;  1315     1878  6                     then begin
;  1316     1879  6                          if  .lnk$gl_ctlmsk [lnk$v_picimg]  !  AND IT'S A PIC IMAGE
;  1317     1880  7                          then begin
;  1318     1881  7                               if .lnk$gl_curisd eql 0       ! IF NO ISECT
;  1319     1882  7                               then true
;  1320     1883  8                               else begin
;  1321     1884  8                                    bind   curhdrisd = lnk$gl_curisd [isl$t_hdrisd] : block [,byte] ;
;  1322     1885  8
;  1323     1886  8                                    if         .curhdrisd [isd$v_wrt]          ! IF SECTION IS WRITEABLE
;  1324     1887  8                                       and not .curhdrisd [isd$v_crf]     !  BUT NOT COPY ON REF (IE SHAREABLE
;  1325     1888  9                                    then  begin
;  1326     1889  9                                          baseallimages () ;                  ! BASE THE WORLD
;  1327     1890  9                                          signal  (lin$_adrwrtdat, 4          ! TELL USER IMAGE ABOUT .ADD
;  1328     1891  9                                                  ,lnk$gl_curpsc [psc$b_namlng]
;  1329     1892 10                                                  ,(.lnk$gl_locn-.curpsectbase)
;  1330     1893  9                                                  ,obmodesc [omd$b_namlng]
;  1331     1894  9                                                  ,lnk$gl_curfil [fdb$q_filename]
;  1332     1895  9                                                  ) ;
;  1333     1896  9                                          false
;  1334     1897  9                                          end
```

```
;1335   1898  8                    else true
;1336   1899  8                    end
;1337   1900  7                end
;1338   1901  7            else (     not .lnk$gl_ctlmsk [lnk$v_lbased]      ! IF NON-PIC, DON'T DO FIXUP IF WORL
;1339   1902  7                   and not .lnk$gl_ctlmsk [lnk$v_ubased]      ! OR IF BASED FOR ANY OTHER REASON (
;1340   1903  9                   and ( (.shrimgsym neq 0)                   !  IF THAT ISN'T TRUE, ONLY DO FIXUP
;1341   1904  8                             and not .symclu [clu$v_based]    !       ! TARGET CLUSTER IS NOT BASE
;1342   1905  7                   )   )
;1343   1906  6            end
;1344   1907  6        else (     .shrimgsym neq 0                           ! EXECUTABLE IMAGE...DO ONLY IF REFE
;1345   1908  6               and .symclu neq .lnk$gl_curclu                 ! IS TO SYMBOL IN ANOTHER CLUSTER
;1346   1909  6               and .symclu [clu$v_shrimg]                     !  THAT IS A SHAREABLE IMAGE
;1347   1910  6               and not .symclu [clu$v_based]                  !    AND IS NOT BASED
;1348   1911  6               )                                              !% check that (,) removal maintained
;1349   1912  5        end
;1350   1913  5    then begin
;1351   1914  5         if not .stapx                                         ! IF IT WAS NOT STA_PX
;1352   1915  6         and (.shrimgsym neq 0)                                !  IE IT WAS STA_GBL
;1353   1916  6         and (.shrsymexpr eql 0)                               !  AND NOT AN EXPRESSION
;1354   1917  5         and .shrimgsym [sym$v_rel]                            ! OF A RELOCATABLE SYMBOL
;1355   1918  5         and .shrimgsym [sym$v_gref]                           ! FROM A SHAREABLE IMAGE
;1356   1919  5         then value = .shrimgsym [sym$l_offset] ;              !   THEN GET VALUE OF SHR IMG SYMBOL
;1357   1920  5
;1358   1921  5         setaddressref (.symclu,.lnk$gl_curisd,.lnk$gl_locn) ;      ! STORE FIXUP INFORMATION
;1359   1922  5
;1360   1923  5         if  .lnk$gl_ctlmsk [lnk$v_long]                       ! IF GETTING A FULL MAP
;1361   1924  6         then begin                                            !  THEN COUNT THE .ADDRESSES
;1362   1925  6              if .obmodesc [omd$l_adrcnt] eql 0                ! IF THIS IS FIRST .ADDRESS IN THIS
;1363   1926  7              then  begin
;1364   1927  7                    local  lastomd : ref block [,byte] ;
;1365   1928  7
;1366   1929  7                    lnk$gl_nadromd = .lnk$gl_nadromd + 1 ;   ! COUNT ANOTHER MODULE WITH .ADDRESS
;1367   1930  7                    lastomd        = .lnk$gl_omdadrend ;     ! GET POINTER TO LAST MODULE IN LIST
;1368   1931  7
;1369   1932  7                    if  .lastomd eql 0                       ! IF NONE, THEN INSERT AT FRONT OF L
;1370   1933  7                    then lnk$gl_omdadrlst       = .obmodesc
;1371   1934  7                    else lastomd [omd$l_nxtadr] = .obmodesc;! OTHERWISE, INSERT AT END OF LIST
;1372   1935  7
;1373   1936  7                    lnk$gl_omdadrend = .obmodesc ;           ! SET NEW END OF LIST
;1374   1937  6                    end ;
;1375   1938  6                obmodesc [omd$l_adrcnt] = .obmodesc [omd$l_adrcnt] + 1 ; ! COUNT THE .ADDRESS IN T
;1376   1939  5                end ;
;1377   1940  4           end ;
;1378   1941  4      end
;1379   1942  3  end
;1380   1943  2  else
;1381   1944  2      !
;1382   1945  2      !  PROCESS TIR$C_STO_PICR (STORE POS. IND. CODE REF)
;1383   1946  2      !
;1384   1947  3      begin
;1385   1948  4      bind symclu  =  (if .stapx                                ! NAME THE CLUSTER DESCRIPTOR
;1386   1949  4                      then shrimgsym [psc$l_cludsc]
;1387   1950  4                      else shrimgsym [sym$l_cludsc]
;1388   1951  3                     ) : ref block [,byte] ;
;1389   1952  3
;1390   1953  3      outputaddr = outputarray [0] ;                            ! WE WILL BE SENDING A MODE BYTE ALSO
;1391   1954  3      if .pdl eql 0                                            ! IF ABS. ADDRESSING
```

```
;  1392          1955  3          then outputarray [0] = absaddrmode                          !  THEN SET THAT
;  1393          1956  3          else
;  1394          1957  3               !
;  1395          1958  3               ! TARGET IS RELOCATABLE
;  1396          1959  3               !
;  1397          1960  4               begin
;  1398          1961  4               if  .shrimgsym neq 0                                  ! IF TARGET IS IN A SHR IMAGE
;  1399          1962  4                   and
;  1400          1963  4                  .symclu [clu$v_shrimg]
;  1401          1964  5               then begin
;  1402          1965  5                    outputarray [0] = indpcreladdr ;                 ! USE "@L^" MODE
;  1403          1966  5                    if .shrsymexpr neq 0                             ! IF TARGET IS AN EXPRESSION
;  1404          1967  5                    or .stapx                                        !  OR A PSECT BASE + OFFSET
;  1405          1968  6                    then begin
;  1406          1969  6                         local  entry : ref block [,byte] ;
;  1407          1970  6
;  1408          1971  6                         if .symclu [clu$l_spcrlst] eql 0              ! IF THIS IS FIRST ONE FOR THIS CLUSTER
;  1409          1972  6                         then lnk$gl_spcgimgs = .lnk$gl_spcgimgs + 1 ; ! THEN COUNT ANOTHER SHR IMG
;  1410          1973  6
;  1411          1974  6                         if lib$insert_tree (symclu [clu$l_spcrlst]    ! INSERT INTO SPCR LIST FOR
;  1412          1975  6                                            ,.value, %ref (0)          ! TARGET CLUSTER
;  1413          1976  6                                            ,compare_spcr
;  1414          1977  6                                            ,alloc_spcr, entry
;  1415          1978  6                                            )
;  1416          1979  6                             eql lib$_normal                          ! AND IF IT WAS INSERTED (NOT ALREADY THERE)
;  1417          1980  6                         then setaddressref (.symclu, 0               ! THEN SET A .ADDR FIXUP FOR THE LOC. IN FIX
;  1418          1981  6                                            ,.entry [spcr$l_fixadr]
;  1419          1982  6                                            ) ;
;  1420          1983  6                         value = .entry [spcr$l_fixadr] ;             ! VALUE TO STORE IS VALUE IN FIXUP
;  1421          1984  6                         end                                          ! (TARGET IS EXPR OR STA_PX)
;  1422          1985  5                    end                                               ! (TARGET IS IN SHR IMG)
;  1423          1986  4               else outputarray [0] = pcreladdrmode ;               ! NOT SHR IMAGE USE PC RELATIVE
;  1424          1987  4
;  1425          1988  4                    value = .value-.lnk$gl_locn - .outbytecount ;    ! PC RELATIVE (SO COMPUTE DISPLACEMENT
;  1426          1989  4                    end                                              ! (TARGET IS RELOCATABLE)
;  1427          1990  2          end ;
;  1428          1991  2
;  1429          1992  2 shrimgsym = stapx = flags = shrsymexpr = absexpr = 0 ;               ! CLEAR FLAGS
;  1430          1993  2 return lnk$wrtimgbyts (.outbytecount,.outputaddr,.repeatcount) ;! FINALLY OUTPUT BYTES
;  1431          1994  1 end ;


                                    52  DD 00000 STORECOMMAND:
                                                PUSHL   R2                                      ; 1670
                              5E            1C  C2 00002          SUBL2   #28, SP                 ;
                              50            5B  9A 00005          MOVZBL  COMMANDTYPE, STOCMDINDEX ; 1690
                              50            14  C2 00008          SUBL2   #20, STOCMDINDEX        ;
                                  F659 CF40 3F  00008          PUSHAW  STORECTLTBL+1[STOCMDINDEX] ; 1691
                              55            9E  9A 00010          MOVZBL  @(SP)+, OUTBYTECOUNT    ;
                              57      F650 CF40 3E  00013          MOVAW   STORECTLTBL[STOCMDINDEX], R7 ; 1693
                              14            67  E9 00019          BLBC    (R7), 2$                ; 1696
                                 00000000'  EF  9F 0001C          PUSHAB  PDL                     ; 1698
                                       08   AE  9F 00022          PUSHAB  REPEATCOUNT             ;
                        F92D  CF            02  FB 00025          CALLS   #2, POP                 ;
```

```
                07            50  E8 0002A          BLBS    R0, 3$                                          ;
                          05B9  31 0002D 1$:       BRW     66$                                             ; 1699
          04    AE            01  D0 00030 2$:       MOVL    #1, REPEATCOUNT                                 ; 1701
                29            5B  91 00034 3$:       CMPB    COMMANDTYPE, #41                                ; 1703
                              3B  12 00037           BNEQ    5$                                              ;
56 00000000'    EF 00000000'  EF  C1 00039           ADDL3   TIROFFSET, TIRREC, R6                           ; 1705
                              56  D6 00045           INCL    R6                                              ;
                55            66  9A 00047           MOVZBL  (R6), OUTBYTECOUNT                              ; 1707
                59        01  A5  9E 0004A           MOVAB   1(R5), DATALNG                                  ; 1708
                    00000000' EF  94 0004E           CLRB    STAPX                                           ; 1709
                    00000000' EF  7C 00054           CLRQ    SHRIMGSYM                                       ;
                50        01 A645 9E 0005A           MOVAB   1(R6)[OUTBYTECOUNT], R0                         ; 1711
   00000000'    EF            50  D1 0005F           CMPL    R0, ENDRECADDR                                  ;
                              03  1B 00066           BLEQU   4$                                              ;
                          01FE  31 00068           BRW     23$                                             ;
                    04    AE    DD 0006B 4$:       PUSHL   REPEATCOUNT                                     ; 1714
                    01    A6    9F 0006E           PUSHAB  1(R6)                                           ; 1713
                          056A  31 00071           BRW     65$                                             ;
                    00000000' EF  9F 00074 5$:       PUSHAB  PDL                                             ; 1719
                    19    AE    9F 0007A           PUSHAB  VALUE                                           ;
          F8D5  CF            02  FB 0007D           CALLS   #2, POP                                         ;
                A8            50  E9 00082           BLBC    R0, 1$                                          ;
                    00000000' EF  D5 00085           TSTL    PDL                                             ; 1722
                              1E  12 0008B           BNEQ    8$                                              ;
                03 00000000' EF  E9 0008D           BLBC    STAPX, 7$                                       ; 1724
                          00C3  31 00094 6$:       BRW     14$                                             ;
                    00000000' EF  D5 00097 7$:       TSTL    SHRIMGSYM                                       ;
                              F5  13 0009D           BEQL    6$                                              ;
                50 00000000' EF  D0 0009F           MOVL    SHRIMGSYM, R0                                   ;
E9        0B    A0            06  E1 000A6           BBC     #6, 11(R0), 6$                                  ;
                1C            5B  91 000AB 8$:       CMPB    COMMANDTYPE, #28                                ; 1726
                              E4  13 000AE           BEQL    6$                                              ;
                1B            5B  91 000B0           CMPB    COMMANDTYPE, #27                                ; 1727
                              DF  13 000B3           BEQL    6$                                              ;
D7 00000000G    00            03  E0 000B5           BBS     #3, LNK$GL_CTLMSK+3, 6$                         ; 1728
                    00000000' EF  DD 000BD           PUSHL   SHRIMGSYM                                       ; 1730
          FDD7  CF            01  FB 000C3           CALLS   #1, CHECKSHRSYM                                 ;
                1D            50  E8 000C8           BLBS    R0, 11$                                         ;
                0E 00000000'  EF  E8 000CB           BLBS    STAPX, 9$                                       ; 1731
50 00000000'    FF 00000000'  EF  C1 000D2           ADDL3   ABSEXPR, @SHRIMGSYM, R0                         ; 1732
                              02  11 000DE           BRB     10$                                             ;
                              50  D4 000E0 9$:       CLRL    R0                                              ; 1731
             15   AE            50  D0 000E2 10$:      MOVL    R0, VALUE                                       ;
                              60  11 000E6           BRB     13$                                             ;
                01 00000000' EF  D1 000E8 11$:      CMPL    PDL, #1                                         ; 1736
                              57  12 000EF           BNEQ    13$                                             ;
4F 00000000G    00            02  E1 000F1           BBC     #2, LNK$GL_CTLMSK, 13$                          ; 1737
47 00000000G    00            03  E0 000F9           BBS     #3, LNK$GL_CTLMSK+3, 13$                        ; 1738
                2A            5B  91 00101           CMPB    COMMANDTYPE, #42                                ; 1739
                              42  13 00104           BEQL    13$                                             ;
                17            5B  91 00106           CMPB    COMMANDTYPE, #23                                ; 1740
                              05  1F 00109           BLSSU   12$                                             ;
                19            5B  91 0010B           CMPB    COMMANDTYPE, #25                                ; 1742
                              38  1B 0010E           BLEQU   13$                                             ;
          FCCB  CF            00  FB 00110 12$:      CALLS   #0, BASEALLIMAGES                               ; 1745
7E 00000000G    00            14  C1 00115           ADDL3   #20, LNK$GL_CURFIL, -(SP)                       ; 1749
7E 00000000'    EF            28  C1 0011D           ADDL3   #40, OBMODESC, -(SP)                            ; 1748
```

```
                7E 00000000G   00 00000000'   EF   C3 00125          SUBL3   CURPSECTBASE, LNK$GL_LOCN, -(SP)          ; 1747
                7E 00000000'   EF               2D   C1 00131          ADDL3   #45, LNK$GL_CURPSC, -(SP)          ; 1746
                                                04   DD 00139          PUSHL   #4                                 ; 1749
                                   00000000G    8F   DD 0013B          PUSHL   #LIN$_NONPICREF                    ;
                   00000000G    00               06   FB 00141          CALLS   #6, LIB$SIGNAL                     ;
                                   00000000'    EF   D4 00148 13$:     CLRL    SHRSYMEXPR                         ; 1752
                                   00000000'    EF   7C 0014E          CLRQ    ABSEXPR                            ;
                                   00000000'    EF   94 00154          CLRB    STAPX                              ;
                60                 67           01   E1 0015A 14$:     BBC     #1, (R7), 16$                      ; 1755
                                   00000000'    EF   D5 0015E          TSTL    PDL                                ; 1757
                                                4A   12 00164          BNEQ    15$                                ;
                42 00000000G    00              02   E1 00166          BBC     #2, LNK$GL_CTLMSK, 15$             ; 1758
                3A 00000000G    00              01   E1 0016E          BBC     #1, LNK$GL_CTLMSK+2, 15$           ; 1759
                                   33 00000000'  EF   E8 00176          BLBS    FLAGS, 15$                         ; 1760
                7E 00000000G    00              14   C1 0017D          ADDL3   #20, LNK$GL_CURFIL, -(SP)          ; 1764
                7E 00000000'    EF              28   C1 00185          ADDL3   #40, OBMODESC, -(SP)               ; 1763
                7E 00000000G    00 00000000'    EF   C3 0018D          SUBL3   CURPSECTBASE, LNK$GL_LOCN, -(SP)   ; 1762
                7E 00000000'    EF              2D   C1 00199          ADDL3   #45, LNK$GL_CURPSC, -(SP)          ; 1761
                                                04   DD 001A1          PUSHL   #4                                 ; 1764
                                   00000000G    8F   DD 001A3          PUSHL   #LIN$_STDISPIC                     ;
                   00000000G    00               06   FB 001A9          CALLS   #6, LIB$SIGNAL                     ;
                50        15   AE 00000000G    00   C3 001B0 15$:     SUBL3   LNK$GL_LOCN, VALUE, R0             ; 1767
          15    AE              50               55   C3 001B9          SUBL3   OUTBYTECOUNT, R0, VALUE            ;
     50         67              05               03   EF 001BE 16$:     EXTZV   #3, #5, (R7), LOWESTMBZBIT         ; 1770
                0A              67               02   E1 001C3          BBC     #2, (R7), 17$                      ; 1775
     51    15   AE              01               50   EF 001C7          EXTZV   LOWESTMBZBIT, #1, VALUE, R1        ; 1776
                                                02   13 001CD          BEQL    17$                                ;
                                                50   D4 001CF          CLRL    LOWESTMBZBIT                       ; 1777
                                                50   D5 001D1 17$:     TSTL    LOWESTMBZBIT                       ; 1779
                                                14   12 001D3          BNEQ    19$                                ;
                50              55               03   78 001D5          ASHL    #3, OUTBYTECOUNT, R0               ; 1780
                                20               50   D1 001D9          CMPL    R0, #32                            ;
                                                03   15 001DC          BLEQ    18$                                ;
                                50               20   D0 001DE          MOVL    #32, R0                            ;
     56    15   AE              50               00   EE 001E1 18$:     EXTV    #0, R0, VALUE, CORRECTVALUE        ;
                                                0E   11 001E7          BRB     20$                                ;
                50              01               50   78 001E9 19$:     ASHL    LOWESTMBZBIT, #1, R0               ; 1782
                                                50   D7 001ED          DECL    MBZMASK                            ;
                                56               50   D2 001EF          MCOML   MBZMASK, CORRECTVALUE              ; 1783
                56        15    AE              56   CB 001F2          BICL3   CORRECTVALUE, VALUE, CORRECTVALUE   ;
                                56        15     AE   D1 001F7 20$:     CMPL    VALUE, CORRECTVALUE                ; 1786
                                                49   13 001FB          BEQL    21$                                ;
                                50 00000000G    00   D0 001FD          MOVL    LNK$GL_LOCN, R0                    ; 1795
                                                50   DD 00204          PUSHL   R0                                 ;
                                                56   DD 00206          PUSHL   CORRECTVALUE                       ; 1794
                                          1D    AE   DD 00208          PUSHL   VALUE                              ;
                                                03   DD 0020B          PUSHL   #3                                 ; 1792
                                   00000000G    8F   DD 0020D          PUSHL   #LIN$_TRUNCDAT                     ;
                7E 00000000G    00              14   C1 00213          ADDL3   #20, LNK$GL_CURFIL, -(SP)          ;
                7E 00000000'    EF              28   C1 0021B          ADDL3   #40, OBMODESC, -(SP)               ; 1791
                7E                 50 00000000' EF   C3 00223          SUBL3   CURPSECTBASE, R0, -(SP)            ; 1790
                7E 00000000'    EF              2D   C1 0022B          ADDL3   #45, LNK$GL_CURPSC, -(SP)          ; 1789
                                                04   DD 00233          PUSHL   #4                                 ; 1792
                                   00000000G    8F   DD 00235          PUSHL   #LIN$_TRUNC                        ;
                   00000000G    00               0B   FB 0023B          CALLS   #11, LIB$SIGNAL                    ;
                         15     AE               56   D0 00242          MOVL    CORRECTVALUE, VALUE                ; 1797
                                20               5B   91 00246 21$:     CMPB    COMMANDTYPE, #32                   ; 1800
```

```
                                    03  13 00249          BEQL    22$                                     ;
                                0095    31 0024B          BRW     29$                                     ;
                            59      02  D0 0024E 22$:     MOVL    #2, DATALNG                             ; 1805
50 00000000'    EF 00000000'        EF  C1 00251          ADDL3   TIROFFSET, TIRREC, R0                   ; 1806
                            50      02  C0 0025D          ADDL2   #2, R0
   00000000'    EF                  50  D1 00260          CMPL    R0, ENDRECADDR                          ;
                                    07  1B 00267          BLEQU   24$                                     ;
       F45B     CF                  00  FB 00269 23$:     CALLS   #0, REPORT_TIRLNG                       ; 1808
                                    70  11 0026E          BRB     28$                                     ;
50 00000000'    EF 00000000'        EF  C1 00270 24$:     ADDL3   TIROFFSET, TIRREC, R0                   ; 1810
                            50      01  A0 9A 0027C       MOVZBL  1(R0), VPSPOS                           ;
51 00000000'    EF 00000000'        EF  C1 00280          ADDL3   TIROFFSET, TIRREC, R1                   ; 1811
                            56      02  A1 9A 0028C       MOVZBL  2(R1), VPSSIZ                           ;
57                          50          56  C1 00290      ADDL3   VPSSIZ, VPSPOS, R7                      ; 1813
                            20          57  D1 00294      CMPL    R7, #32                                 ;
                                        0-  1E 00297      BGEQU   25$                                     ;
                            51      08  A0 9E 00299       MOVAB   8(R0), R1                               ; 1814
                            51          08  C6 0029D      DIVL2   #8, R1
                            51          08  C4 002A0      MULL2   #8, R1
                            51          57  D1 002A3      CMPL    R7, R1                                  ;
                                        26  1B 002A6      BLEQU   27$                                     ;
7E 00000000G    00                      14  C1 002A8 25$:     ADDL3   #20, LNK$GL_CURFIL, -(SP)         ; 1818
7E 00000000'    EF                      28  C1 002B0          ADDL3   #40, OBMODESC, -(SP)              ; 1817
                                0041    8F  BB 002B8          PUSHR   #^M<R0,R6>                        ; 1818
                                        04  DD 002BC          PUSHL   #4                                ;
                            00000000G   8F  DD 002BE          PUSHL   #LIN$_ILLVPS                      ;
   00000000G    00                      06  FB 002C4          CALLS   #6, LIB$SIGNAL                    ;
                                    031B    31 002CB 26$:     BRW     66$                               ; 1822
                                0041    8F  BB 002CE 27$:     PUSHR   #^M<R0,R6>                        ;
                                        01  DD 002D2          PUSHL   #1                                ;
                                  21    AE  DD 002D4          PUSHL   VALUE                             ;
                                        01  DD 002D7          PUSHL   #1                                ;
   00000000G    00                      05  FB 002D9          CALLS   #5, LNK$WRTIMGBYTS                ;
                                    0308    31 002E0 28$:     BRW     67$                               ;
                            2A              5B  91 002E3 29$:     CMPB    COMMANDTYPE, #42              ; 1825
                                            03  13 002E6          BEQL    30$                           ;
                                        00C9    31 002E8          BRW     38$                           ;
                                    08      AE  9F 002EB 30$:     PUSHAB  PDL2                          ; 1830
                                    10      AE  9F 002EE          PUSHAB  VALUE2                        ;
       F661     CF                      02  FB 002F1          CALLS   #2, POP                           ;
                            D2              50  E9 002F6      BLBC    R0, 26$                           ;
                                    08      AE  D5 002F9      TSTL    PDL2                              ; 1833
                                            1B  12 002FC      BNEQ    31$                               ;
                            40 00000000'        EF  E8 002FE      BLBS    STAPX, 34$                    ; 1835
                               00000000'        EF  D5 00305      TSTL    SHRIMGSYM                     ;
                                            38  13 0030B      BEQL    34$                               ;
                            50 00000000'        EF  D0 0030D      MOVL    SHRIMGSYM, R0                 ;
2C          0B  A0                          06  E1 00314      BBC     #6, 11(R0), 34$                   ;
74 00000000G    00                          03  E0 00319 31$:     BBS     #3, LNK$GL_CTLMSK+3, 36$      ; 1837
                               00000000'        EF  DD 00321      PUSHL   SHRIMGSYM                     ; 1839
       FB73     CF                          01  FB 00327      CALLS   #1, CHECKSHRSYM                   ;
                            18              50  E8 0032C      BLBS    R0, 35$                           ;
                            09 00000000'        EF  E8 0032F      BLBS    STAPX, 32$                    ; 1840
                            50 00000000'        FF  D0 00336      MOVL    @SHRIMGSYM, R0                ; 1841
                                            02  11 0033D      BRB     33$                               ;
                                            50  D4 0033F 32$:     CLRL    R0                            ; 1840
         0C     AE                          50  D0 00341 33$:     MOVL    R0, VALUE2                    ;
```

```
                                 4E  11 00345 34$:     BRB     36$
                          01        08   AE  D1 00347 35$:     CMPL    PDL2, #1                                        ; 1844
                                 48  12 00348             BNEQ    36$
40 00000000C              00        02   E1  00034D             BBC     #2, LNK$GL_CTLMSK, 36$                          ; 1845
38 00000000G              00        03   E0  00355             BBS     #3, LNK$GL_CTLMSK+3, 36$                        ; 1846
                             FA7E   CF   00  FB  0035D             CALLS   #0, BASEAL[IMAGES                               ; 1848
7E 00000000G              00        14   C1  00362             ADDL3   #20, LNK$GL_CURFIL, -(SP)                       ; 1853
7E 00000000'              EF        28   C1  0036A             ADDL3   #40, OBMODESC, -(SP)                            ; 1852
7E 00000000G              00 00000000'   EF  C3  00372             SUBL3   CURPSECTBASE, LNK$GL_LOCN, -(SP)                ; 1851
7E 00000000'              EF        2D   C1  0037E             ADDL3   #45, LNK$GL_CURPSC, =(SP)                       ; 1850
                                 04  DD  00386             PUSHL   #4                                                      ; 1853
                             00000000G   8F  DD  00388             PUSHL   #LIN$_NONPICREF
                00000000G    00        06   FB  0038E             CALLS   #6, LIB$SIGNAL
                FFFFFFFF     8F        0C   AE  D1 00395 36$:     CMPL    VALUE2, #-1                                     ; 1858
                                 08  12 0039D             BNEQ    37$
                      0C     AE 00000000G   00  D0  0039F             MOVL    LNK$GL_LOCN, VALUE2
                             00000000'   EF  D5  003A7 37$:     TSTL    PDL                                             ; 1859
                                 05  13 003AD             BEQL    38$
                      15     AE        0C   AE  C2  003AF             SUBL2   VALUE2, VALUE
                                 1C  5B 91 003B4 38$:     CMPB    COMMANDTYPE, #28                                ; 1862
                                 0C  13 003B7             BEQL    40$
                                 1B  5B 91 003B9             CMPB    COMMANDTYPE, #27                                ; 1863
                                 07  13 003BC             BEQL    40$
                                 58        15   AE  9E  003BE             MOVAB   OUTPUTARRAY+1, OUTPUTADDR               ; 1864
                                    01FC  31  003C2 39$:     BRW     64$
                                 52 00000000'   EF  9A  003C5 40$:     MOVZBL  STAPX, R2                               ; 1867
                                 51 00000000'   EF  D0  003CC             MOVL    SHRIMGSYM, R1                           ; 1868
                                 57 00000000'   EF  D0  003D3             MOVL    PDL, R7                                 ; 1873
                                 1B  5B 91 003DA             CMPB    COMMANDTYPE, #27                                ; 1865
                                 03  13 003DD             BEQL    41$
                                    0145  31  003DF             BRW     54$
                                 06        52   E9  003E2 41$:     BLBC    R2, 42$                                         ; 1867
                                 50        24   A1  9E  003E5             MOVAB   36(R1), R0                              ; 1868
                                 04  11 003E9             BRB     43$
                                 50        20   A1  9E  003EB 42$:     MOVAB   32(R1), R0                              ; 1869
                                 56        50   D0  003EF 43$:     MOVL    R0, R6
                                 58        15   AE  9E  003F2             MOVAB   OUTPUTARRAY+1, OUTPUTADDR               ; 1872
                                 57  D5 003F6             TSTL    R7                                                      ; 1873
                                 C8  13 003F8             BEQL    39$
                                 C1 00000000G   00  E8  003FA             BLBS    LNK$GL_CTLMSK+2, 39$                    ; 1874
79 00000000G                  00        02   E1  00401             BBC     #2, LNK$GL_CTLMSK, 47$                          ; 1877
53 00000000G                  00        01   E1  00409             BBC     #1, LNK$GL_CTLMSK+2, 45$                        ; 1879
                                 50 00000000G   00  D0  00411             MOVL    LNK$GL_CURTSD, R0                       ; 1881
                                 03  12 00418             BNEQ    44$
                                    0081  31  0041A             BRW     49$
                                 50        18   C0  0041D 44$:     ADDL2   #24, R0                                         ; 1884
79                  08        A0        03   E1  00420             BBC     #3, 8(R0), 49$                                  ; 1886
74                  08        A0        01   E0  00425             BBS     #1, 8(R0), 49$                                  ; 1887
                       F9B1   CF   00  FB  0042A             CALLS   #0, BASEALLIMAGES                               ; 1889
7E 00000000G              00        14   C1  0042F             ADDL3   #20, LNK$GL_CURFIL, -(SP)                       ; 1894
7E 00000000'              EF        28   C1  00437             ADDL3   #40, OBMODESC, -(SP)                            ; 1893
7E 00000000G              00 00000000'   EF  C3  0043F             SUBL3   CURPSECTBASE, LNK$GL_LOCN, -(SP)                ; 1892
7E 00000000'              EF        2D   C1  0044B             ADDL3   #45, LNK$GL_CURPSC, =(SP)                       ; 1891
                                 04  DD  00453             PUSHL   #4                                                      ; 1894
                             00000000G   8F  DD  00455             PUSHL   #LIN$_ADRWRTDAT
                00000000G    00        06   FB  0045B             CALLS   #6, LIB$SIGNAL
                                 1B  11 00462             BRB     46$
```

```
                 13 00000000G   00            04   E0 00464 45$:    BBS     #4, LNK$GL_CTLMSK+3, 46$                    ; 1901
                 0B 00000000G   00            03   E0 0046C          BBS     #3, LNK$GL_CTLMSK+3, 46$                    ; 1902
                                              51   D5 00474          TSTL    R1                                          ; 1903
                                              15   13 00476          BEQL    48$
                                    50        66   D0 00478          MOVL    (R6), R0                                    ; 1904
                                    1F    58  A0   E9 0047B          BLBC    88(R0), 49$
                                          013F  31   0047F 46$:       BRW     64$
                                              51   D5 00482 47$:     TSTL    R1                                          ; 1907
                                              F9   13 00484          BEQL    46$
                        00000000G   00            66   D1 00486          CMPL    (R6), LNK$GL_CURCLU                         ; 1908
                                              F0   13 0048D 48$:     BEQL    46$
                                    50        66   D0 0048F          MOVL    (R6), R0                                    ; 1909
                 E8         58    A0            02   E1 00492          BBC     #2, 88(R0), 46$
                                    50        66   D0 00497          MOVL    (R6), R0                                    ; 1910
                                    E1    58  A0   E8 0049A          BLBS    88(R0), 46$
                                    34 00000000'  EF   E8 0049E 49$:  BLBS    STAPX, 50$                                  ; 1914
                                       00000000'  EF   D5 004A5       TSTL    SHRIMGSYM                                   ; 1915
                                                  2C   13 004AB       BEQL    50$
                                       00000000'  EF   D5 004AD       TSTL    SHRSYMEXPR                                  ; 1916
                                                  24   12 004B3       BNEQ    50$
                                    50 00000000'  EF   D0 004B5       MOVL    SHRIMGSYM, R0                               ; 1917
                 18         0A    A0            03   E1 004BC          BBC     #3, 10(R0), 50$
                                    50 00000000'  EF   D0 004C1       MOVL    SHRIMGSYM, R0                               ; 1918
                 0C         0B    A0            06   E1 004C8          BBC     #6, 11(R0), 50$
                                    50 00000000'  EF   D0 004CD       MOVL    SHRIMGSYM, R0                               ; 1919
                                    15    AE  10   A0   D0 004D4       MOVL    16(R0), VALUE
                                       00000000G  00   DD 004D9 50$:  PUSHL   LNK$GL_LOCN                                 ; 1921
                                       00000000G  00   DD 004DF       PUSHL   LNK$GL_CURISD
                                                  66   DD 004E5       PUSHL   (R6)
                                          FA49    CF   03   FB 004E7       CALLS   #3, SETADDRESSREF
                                    8C 00000000G  00   E9 004EC       BLBC    LNK$GL_CTLMSK+1, 46$                        ; 1923
                                    50 00000000'  EF   D0 004F3       MOVL    OBMODESC, R0                                ; 1925
                                              24   A0   D5 004FA       TSTL    36(R0)
                                                   23   12 004FD       BNEQ    53$
                                       00000000'  EF   D6 004FF       INCL    LNK$GL_NADROMD                              ; 1929
                                    51 00000000'  EF   D0 00505       MOVL    LNK$GL_OMDADREND, LASTOMD                   ; 1930
                                                  09   12 0050C       BNEQ    51$                                         ; 1932
                        00000000'   EF            50   D0 0050E       MOVL    R0, LNK$GL_OMDADRLST                        ; 1933
                                                  04   11 00515       BRB     52$
                                20  A1            50   D0 00517 51$:  MOVL    R0, 32(LASTOMD)                             ; 1934
                        00000000'   EF            50   D0 0051B 52$:  MOVL    R0, LNK$GL_OMDADREND                        ; 1936
                                              24   A0   D6 00522 53$:  INCL    36(R0)                                      ; 1938
                                                   1A   11 00525       BRB     57$                                         ; 1873
                                    06            52   E9 00527 54$:  BLBC    R2, 55$                                     ; 1948
                                    50        24  A1   9E 0052A       MOVAB   36(R1), R0                                  ; 1949
                                                  04   11 0052E       BRB     56$
                                    50        20  A1   9E 00530 55$:  MOVAB   32(R1), R0                                  ; 1950
                                    58        14  AE   9E 00534 56$:  MOVAB   OUTPUTARRAY, OUTPUTADDR                    ; 1953
                                                  57   D5 00538       TSTL    R7                                          ; 1954
                                                  08   12 0053A       BNEQ    58$
                                14  AE        9F   8F   90 0053C       MOVB    #-97, OUTPUTARRAY                           ; 1955
                                          007D  31   00541 57$:       BRW     64$
                                              51   D5 00544 58$:     TSTL    R1                                          ; 1961
                                              67   13 00546          BEQL    62$
                                    56            60   D0 00548          MOVL    (R0), R6                                    ; 1963
                 5F         58    A6            02   E1 0054B          BBC     #2, 88(R6), 62$
                                14  AE            01   8E 00550          MNEGB   #1, OUTPUTARRAY                             ; 1965
```

```
                                00000000'  EF  D5 00554          TSTL    SHRSYMEXPR                                  ; 1966
                                           03  12 0055A          BNEQ    59$
                                54         52  E9 0055C          BLBC    R2, 63$                                     ; 1967
                                56         60  D0 0055F 59$:     MOVL    (R0), R6                                    ; 1971
                                      28   A6  D5 00562          TSTL    40(R6)
                                           06  12 00565          BNEQ    60$
                                00000000'  EF  D6 00567          INCL    LNK$GL_SPCGIMGS                             ; 1972
                                      10   AE  9F 0056D 60$:     PUSHAB  ENTRY                                       ; 1974
                                    FA58   CF  9F 00570          PUSHAB  ALLOC_SPCR
                                    FA47   CF  9F 00574          PUSHAB  COMPARE_SPCR
                                      0C   AE  D4 00578          CLRL    12(SP)                                      ; 1975
                                      0C   AE  9F 0057B          PUSHAB  12(SP)
                                      25   AE  DD 0057E          PUSHL   VALUE
                                      28   A6  9F 00581          PUSHAB  40(R6)                                      ; 1974
                          00000000G   00      06  FB 00584          CALLS   #6, LIB$INSERT_TREE
                          00000000G   8F      50  D1 0058B          CMPL    R0, #LIB$_NORMAL                            ; 1979
                                           10  12 00592          BNEQ    61$
                                50         10  AE  D0 00594          MOVL    ENTRY, R0                                   ; 1981
                                           0E  A0  DD 00598          PUSHL   14(R0)
                                           7E  D4  00059B          CLRL    -(SP)                                       ; 1980
                                           56  DD  00059D          PUSHL   R6
                                    F991   CF      03  FB 00059F          CALLS   #3, SETADDRESSREF
                                50         10  AE  D0 005A4 61$:     MOVL    ENTRY, R0                                   ; 1983
                                      15   AE  0E  A0  D0 005A8          MOVL    14(R0), VALUE
                                           04  11 005AD          BRB     63$                                         ; 1966
                                      14   AE      11  BE 005AF 62$:     MNEGB   #17, OUTPUTARRAY                    ; 1986
                          50          15   AE  00000000G   00  C3 005B3 63$:     SUBL3   LNK$GL_LOCN, VALUE, R0      ; 1988
                    15    AE              50  55  C3 005BC          SUBL3   OUTBYTECOUNT, R0, VALUE
                                00000000'  EF  D4 005C1 64$:     CLRL    SHRSYMEXPR                                  ; 1992
                                00000000'  EF  B4 005C7          CLRW    FLAGS
                                00000000'  EF  94 005CD          CLRB    STAPX
                                00000000'  EF  7C 005D3          CLRQ    ABSEXPR
                                      04   AE  DD 005D9          PUSHL   REPEATCOUNT                                 ; 1993
                                           58  DD 005DC          PUSHL   OUTPUTADDR
                                           55  DD 005DE 65$:     PUSHL   OUTBYTECOUNT
                          00000000G   00      03  FB 005E0          CALLS   #3, LNK$WRTIMGBYTS
                                           02  11 005E7          BRB     67$
                                           50  D4 005E9 66$:     CLRL    R0                                          ; 1994
                                5E         1C  C0 005EB 67$:     ADDL2   #28, SP
                                           04  BA 005EE          POPR    #^M<R2>
                                           05  005F0          RSB

; Routine Size:  1521 bytes,    Routine Base:  $CODE$ + 09AC

; 1432          1995  1
```

```
; 1434          1996  1 routine operatecommand : tir_proc =
; 1435          1997  2 begin
; 1436          1998  2 !
; 1437          1999  2 ! PROCESS OPERATE GROUP COMMANDS
; 1438          2000  2 !
; 1439          2001  2 external register
; 1440          2002  2     commandtype = 11 : byte,
; 1441          2003  2     commandata = 10 : ref block[,byte],
; 1442          2004  2     datalng = 9;
; 1443          2005  2
; 1444          2006  2 builtin
; 1445          2007  2     rot;
; 1446          2008  2
; 1447          2009  2 local
; 1448          2010  2     first_abs,
; 1449          2011  2     second_abs,
; 1450          2012  2     either_shr,
; 1451          2013  2     cludesc : ref block[,byte],
; 1452          2014  2     symdesc : ref block[,byte],
; 1453          2015  2     symdscnam : ref block[,byte],       ! NAME PART OF BLOCK
; 1454          2016  2     value,
; 1455          2017  2     value2,
; 1456          2018  2     pdl2;
; 1457          2019  2
; 1458          2020  2 if .commandtype eql tir$c_opr_nop
; 1459          2021  2     then return true;
; 1460          2022  2 if .commandtype eql tir$c_opr_redef                  ! IF THIS IS A RE-DEFINITION COMMAND
; 1461          2023  3 then begin                                          ! FIRST VALIDATE THE TIR STREAM
; 1462          2024  3     datalng = 1 + .commandata[symboleng];           ! SET LENGTH OF DATA
; 1463          2025  3     if (commandata[symboleng] +.datalng) gtru .endrecaddr ! CHECK ALL THERE
; 1464          2026  3         then return report_tirlng();
; 1465          2027  3     if .commandata[symboleng] gtru sym$c_maxlng      ! CHECK LENGTH OF SYMBOL NAME
; 1466          2028  3     or .commandata[symboleng] eql 0
; 1467          2029  3     then return report_illnamlng(commandata[symboleng]);
; 1468          2030  3     if not .lnk$gl_ctlmsk[lnk$v_shr]                ! IF NOT PRODUCING A SHAREABLE IMAGE
; 1469          2031  3     then return true;                               ! FORGET IT
; 1470          2032  3     if not lnk$search(commandata[symboleng],symdesc,symdscnam) ! LOOK IT UP
; 1471          2033  3     or not .symdesc[sym$v_def]                      ! CHECK IT FOR A DEFINITION
; 1472          2034  4     then begin                                      ! AND ISSUE ERROR MESSAGE IF NOT
; 1473          2035  4         report_useundef(commandata[symboleng]);
; 1474          2036  4         flags[flg$v_undef] = true ;
; 1475          2037  4         return true                                 . AND FORGET IT
; 1476          2038  3         end;
; 1477          2039  3     if .symdesc[sym$v_redef]                        ! IF THE SYMBOL HAS A REDEF PENDING
; 1478          2040  4     then begin
; 1479          2041  4         signal(lin$_alredef,5,commandata[symboleng],
; 1480          2042  4                 lnk$gl_curpsc[psc$b_namlng],(.lnk$gl_locn-.curpsectbase),
; 1481          2043  4                 obmodesc[omd$b_namlng],lnk$gl_curfil[fdb$q_filename]);
; 1482          2044  4         return true
; 1483          2045  3         end;
; 1484          2046  3     symdesc[sym$l_newval] = .lnk$gl_locn;           ! OTHERWISE SET ITS NEW VALUE
; 1485          2047  3     symdesc[sym$v_redef] = true;                    ! SET REDEF PENDING
; 1486          2048  3     symdesc[sym$v_uni] = true;                      ! AND MAKE IT UNIVERSAL
; 1487          2049  3     cludesc = .lnk$gl_curpsc[psc$l_cludsc];         ! GET POINTER TO CLUSTER DESCRIPTOR
; 1488          2050  3     if .lnk$gl_curpsc[psc$v_rel]                    ! IF CURRENT PSECT IS RELOCATABLE
; 1489          2051  3         and not .cludesc[clu$v_based]               !  IN A CLUSTER THAT IS NOT BASED
; 1490          2052  3         then symdesc[sym$v_rerel] = true;           !  THEN MAKE SURE THAT SYMBOL IS RELOCATABLE (DONE I
```

```
; 1491          2053  3     if .lnk$gl_ctlmsk[lnk$v_mapopn]                          ! IF THERE WAS A MAP OPENED
; 1492          2054  3         and not .lnk$gl_ctlmsk[lnk$v_brief]                  ! AND IT WAS NOT A BRIEF
; 1493          2055  3         then crf$insrtkey(lnk$al_sytblfmt,symdscnam[snb$b_naming], ! TELL CREF ABOUT NEW FLAGS
; 1494          2056  3                           symdesc[sym$l_value],,symdesc[sym$w_flags]);
; 1495          2057  3     return true
; 1496          2058  2     end;                                                     ! THAT'S ALL FOR REDEF
; 1497          2059  2 first_abs = true;
; 1498          2060  2 if not pop(value,pdl)                                       ! FOR ALL OTHERS, GET TOP VALUE FROM STACK
; 1499          2061  2     then return false;
; 1500          2062  2 if .shrimgsym neq 0
; 1501          2063  2 then
; 1502          2064  3     begin
; 1503          2065  3     first_abs = false;
; 1504          2066  3     if not .stapx
; 1505          2067  3         and .shrimgsym[sym$v_gref]
; 1506          2068  3         and .shrimgsym[sym$v_rel]
; 1507          2069  3     then
; 1508          2070  4         begin
; 1509          2071  4         value = .shrimgsym[sym$l_offset];
; 1510          2072  4         shrsymexpr = .shrimgsym;
; 1511          2073  3         end;
; 1512          2074  2     end;
; 1513          2075  2 if .commandtype neq tir$c_opr_neg                          ! POP SECOND VAL E IF NOT NEGATE
; 1514          2076  2     and .commandtype neq tir$c_opr_com                    !  COMPLEMENT
; 1515          2077  2     and .commandtype neq tir$c_opr_dflit                  !  OR DEFINE LITERAL
; 1516          2078  3 then begin
; 1517          2079  3     second_abs = true;
; 1518          2080  3     if not pop(value2,pdl2)                              ! GET SECOND VALUE IF A BINARY
; 1519          2081  3         then return false;                               !  OPERATOR - ERROR IF NONE
; 1520          2082  3     if .shrimgsym neq 0                                  ! IF FROM SHR IMG
; 1521          2083  3     then
; 1522          2084  4         begin
; 1523          2085  4         if .shrsymexpr neq 0                             ! AND OTHER OPERAND WAS TOO
; 1524          2086  4             then signal(lin$_illshrxpr,3,                !  THEN THAT IS AN ERROR
; 1525          2087  4                         obmodesc[omd$b_naming],.lnk$gl_record,
; 1526          2088  4                         lnk$gl_curfil[fdb$q_filename]);
; 1527          2089  4         if not .stapx                                    ! IF STA_GBL
; 1528          2090  4             and .shrimgsym[sym$v_gref]                   ! OF SHR IMG SYMBOL
; 1529          2091  4             and .shrimgsym[sym$v_rel]
; 1530          2092  4         then
; 1531          2093  5             begin
; 1532          2094  5             value2 = .shrimgsym[sym$l_offset];           ! THEN USE THE OFFSET
; 1533          2095  5             shrsymexpr = .shrimgsym;                     ! AND REMEMBER THAT IT WAS
; 1534          2096  4             end;
; 1535          2097  4         second_abs = false;
; 1536          2098  3         end;
; 1537          2099  2     end;
; 1538          2100  2
; 1539          2101  2 if .shrsymexpr neq 0                                     ! IF ONE OPERAND IS FROM SHR IMG
; 1540          2102  2     and .commandtype neq tir$c_opr_add                    ! ONLY ADD AND SUB ARE LEGAL
; 1541          2103  2     and .commandtype neq tir$c_opr_sub
; 1542          2104  2     then signal(lin$_illshropr,3,obmodesc[omd$b_naming], ! ANYTHING ELSE IS ERROR
; 1543          2105  2                 .lnk$gl_record,lnk$gl_curfil[fdb$q_filename]);
; 1544          2106  2 either_shr = not (.first_abs and .second_abs);
; 1545          2107  2 !
; 1546          2108  2 ! DISPATCH ON COMMAND
; 1547          2109  2 !
```

```
;  1548          2110  2 case .commandtype from tir$c_opr_add to tir$c_maxoprcod
;  1549          2111  2 of set
;  1550          2112  2     [tir$c_opr_add] :
;  1551          2113  3                     begin
;  1552          2114  3                         if .either_shr                      ! IF THIS EXPRN IS PART OF A SHR IMG EXPRN
;  1553          2115  3                         then
;  1554          2116  4                             begin
;  1555          2117  4                             if .first_abs                   ! IF FIRST VALUE WAS A CONSTANT
;  1556          2118  4                             then                            !   ACCUMULATE IT
;  1557          2119  4                                 absexpr = .absexpr + .value;
;  1558          2120  4                             if .second_abs                  ! IF SECOND VALUE WAS A CONSTANT
;  1559          2121  4                             then                            !   ACCUMULATE IT
;  1560          2122  4                                 absexpr = .absexpr + .value2;
;  1561          2123  3                             end;
;  1562          2124  3                         value = .value + .value2;    ! ADD THE TWO VALUES
;  1563          2125  3                         pdl = .pdl + .pdl2;          ! AND THE TWO POS. DEP. LEVELS
;  1564          2126  3                         true
;  1565          2127  2                         end;
;  1566          2128  2     [tir$c_opr_sub] :
;  1567          2129  3                     begin
;  1568          2130  3                         if .either_shr                      ! IF THIS EXPRN IS PART OF A SHR IMG EXPRN
;  1569          2131  3                         then
;  1570          2132  4                             begin
;  1571          2133  4                             if .first_abs                   ! IF FIRST VALUE WAS A CONSTANT
;  1572          2134  4                             then                            !   ACCUMULATE IT
;  1573          2135  4                                 absexpr = .absexpr - .value;
;  1574          2136  4                             if .second_abs                  ! IF SECOND VALUE WAS A CONSTANT
;  1575          2137  4                             then                            !   ACCUMULATE IT
;  1576          2138  4                                 absexpr = .absexpr - .value2;
;  1577          2139  3                             end;
;  1578          2140  3                         value = .value2 - .value;    ! SUBTRACT THE VALUES AND
;  1579          2141  3                         pdl = .pdl2 - .pdl;          ! AND POSITION DEPENDENCE LEVELS
;  1580          2142  3                         true
;  1581          2143  2                         end;
;  1582          2144  2     [tir$c_opr_neg] :
;  1583          2145  3                     begin
;  1584          2146  3                         value = -.value;
;  1585          2147  3                         true
;  1586          2148  2                         end;
;  1587          2149  2     [tir$c_opr_com] :
;  1588          2150  3                     begin
;  1589          2151  3                         value = -1 - .value;
;  1590          2152  3                         true
;  1591          2153  2                         end;
;  1592          2154  2     [tir$c_opr_mul] :
;  1593          2155  3                     begin
;  1594          2156  3                         value = .value*.value2;
;  1595          2157  3                         true
;  1596          2158  2                         end;
;  1597          2159  2     [tir$c_opr_div] :
;  1598          2160  3                     begin
;  1599          2161  3                         if .value neq 0                 ! CHECK FOR DIVIDE BY ZERO
;  1600          2162  4                             then value = .value2/.value;
;  1601          2163  3                         true
;  1602          2164  2                         end;
;  1603          2165  2     [tir$c_opr_ior] :
;  1604          2166  3                     begin
```

```
;  1605     2167  1                    value = .value or .value2;
;  1606     2168  1                    true
;  1607     2169  1                    end;
;  1608     2170  1        [tir$c_opr_eor] :
;  1609     2171  1                    begin
;  1610     2172  1                    value = (.value or .value2) and
;  1611     2173  1                                        not(.value and .value2);
;  1612     2174  1                    true
;  1613     2175  1                    end;
;  1614     2176  1        [tir$c_opr_and] :
;  1615     2177  1                    begin
;  1616     2178  1                    value = .value and .value2;
;  1617     2179  1                    true
;  1618     2180  1                    end;
;  1619     2181  1        [tir$c_opr_rot] :
;  1620     2182  1                    begin
;  1621     2183  1                    value = rot(.value2, .value);
;  1622     2184  1                    true
;  1623     2185  1                    end;
;  1624     2186  1        [tir$c_opr_ash] :
;  1625     2187  1                    begin
;  1626     2188  1                    value = .value2 ^ .value;
;  1627     2189  1                    true
;  1628     2190  1                    end;
;  1629     2191  1        [tir$c_opr_sel] :
;  1630     2192  1                    begin
;  1631     2193  1                    if .value eql true                  ! IF TOP LONGWORD IS
;  1632     2194  1                        then return true;               ! IS TRUE, ALL DONE
;  1633     2195  1                    value = .value2;                    ! OTHERWISE MOVE VALUE2
;  1634     2196  1                    pdl = .pdl2;                        ! TO VALUE 3, DISCARDING
;  1635     2197  1                    if not pop(value2,pdl2)             ! WHAT EVER WAS THERE
;  1636     2198  1                        then return false;              ! BUT CHECK SOMETHING WAS
;  1637     2199  1                    end;
;  1638     2200  1        [tir$c_opr_dflit] :
;  1639     2201  1                    begin
;  1640     2202  1                    local
;  1641     2203  1                        entry : ref block[,byte],
;  1642     2204  1                        litindex;
;  1643     2205  1
;  1644     2206  1                    datalng = 1;
;  1645     2207  1                    if (.commandata + .datalng) gtru .endrecaddr
;  1646     2208  1                        then return report_tirlng();
;  1647     2209  1                    litindex = .commandata[0,0,8,0];
;  1648     2210  1                    lib$insert_tree(lnk$gl_litbl,.litindex,             !INSERT INTO TREE
;  1649     2211  1                                    %ref(0),compare_lit,alloc_lit,entry);
;  1650     2212  1                    entry[lit$v_pdl] = .pdl;
;  1651     2213  1                    entry[lit$v_stapx] = .stapx;
;  1652     2214  1                    entry[lit$b_index] = .litindex;
;  1653     2215  1                    entry[lit$l_shrsym] = .shrimgsym;
;  1654     2216  1                    entry[lit$l_shrexpr] = .shrsymexpr;
;  1655     2217  1                    entry[lit$l_value] = .value;
;  1656     2218  1                    lnk$gl_litf[gs[.litindex] = true;
;  1657     2219  1                    return true
;  1658     2220  1                    end;
;  1659     2221  1        [inrange] :
;  1660     2222  1                    return report_illtir2();
;  1661     2223  1        tes;
```

```
; 1662          2224  2
; 1663          2225  2 return push(.value,-1,.pdl)
; 1664          2226  2
; 1665          2227  1 end;


                                    5E            1C  C2 00000 OPERATECOMMAND:
                                                                SUBL2   #28, SP                                         ; 1996
                                    32              5B  91 00003        CMPB    COMMANDTYPE, #50                        ; 2020
                                                    62  13 00006        BEQL    6$
                                 41 8F              5B  91 00008        CMPB    COMMANDTYPE, #65                        ; 2022
                                                    03  13 0000C        BEQL    1$
                                                 00EC   31 0000E        BRW     12$
                                    59              6A  9A 00011 1$:    MOVZBL  (COMMANDATA), DATALNG                   ; 2024
                                                    59  D6 00014        INCL    DATALNG
                              50                    5A  59  C1 00016    ADDL3   DATALNG, COMMANDATA, R0                 ; 2025
                                 00000000'          EF  50  D1 0001A    CMPL    R0, ENDRECADDR
                                                        03  1B 00021    BLEQU   2$
                                                     02F1   31 00023    BRW     42$
                                    1F              6A  91 00026 2$:    CMPB    (COMMANDATA), #31                       ; 2027
                                                        04  1A 00029    BGTRU   3$
                                                    6A  95 0002B        TSTB    (COMMANDATA)                            ; 2028
                                                        0A  12 0002D    BNEQ    4$
                                                    5A  DD 0002F 3$:    PUSHL   COMMANDATA                              ; 2029
                                       F06F     CF      01  FB 00031    CALLS   #1, REPORT_ILLNAMLNG
                                                     035A   31 00036    BRW     48$
                              69 00000000G          00  02  E1 00039 4$:    BBC     #2, LNK$GL_CTLMSK, 8$           ; 2030
                                                 04     AE  9F 00041        PUSHAB  SYMDSCNAM                       ; 2032
                                                 0C     AE  9F 00044        PUSHAB  SYMDESC
                                                    5A  DD 00047        PUSHL   COMMANDATA
                                 00000000G          00  03  FB 00049    CALLS   #3, LNK$SEARCH
                                                    09  50  E9 00050    BLBC    R0, 5$
                                                    50      08  AE  D0 00053    MOVL    SYMDESC, R0                     ; 2033
                              10       0A          A0  01  E0 00057    BBS     #1, 10(R0), 7$
                                                    5A  DD 0005C 5$:    PUSHL   COMMANDATA                      ; 2035
                                    F0C8    CF      01  FB 0005E    CALLS   #1, REPORT_USEUNDEF
                                 00000000'          EF  01  88 00063    BISB2   #1, FLAGS                       ; 2036
                                                    3E  11 0006A 6$:    BRB     8$                              ; 2037
                                                    55      08  AE  D0 0006C 7$:    MOVL    SYMDESC, R5             ; 2039
                              37       0B          A5  04  E1 00070    BBC     #4, 11(R5), 9$
                              7E 00000000G          00  14  C1 00075    ADDL3   #20, LNK$GL_CURFIL, -(SP)               ; 2043
                              7E 00000000'          EF  28  C1 0007D    ADDL3   #40, OBMODESC, -(SP)
                              7E 00000000G          00 00000000'  EF  C3 00085    SUBL3   CURPSECTBASE, LNK$GL_LOCN, -(SP)    ; 2042
                              7E 00000000'          EF  2D  C1 00091    ADDL3   #45, LNK$GL_CURPSC, -(SP)
                                                    5A  DD 00099        PUSHL   COMMANDATA                              ; 2043
                                                    05  DD 0009B        PUSHL   #5
                                                 00000000G  8F  DD 0009D    PUSHL   #LIN$_ALREDEF
                                 00000000G          00  07  FB 000A3    CALLS   #7, LIB$SIGNAL
                                                    4E  11 000AA 8$:    BRB     11$                             ; 2044
                                       10       A5 00000000G  00  D0 000AC 9$:    MOVL    LNK$GL_LOCN, 16(R5)     ; 2046
                                       0A       A5      1004  8F  A8 000B4    BISW2   #4100, 10(R5)                ; 2047
                                                50 00000000'  EF  D0 000BA    MOVL    LNK$GL_CURPSC, R0               ; 2049
                                                51      24  A0  D0 000C1    MOVL    36(R0), CLUDESC
                              08       0A          A0  03  E1 000C5    BBC     #3, 10(R0), 10$                 ; 2050
```

```
                    04         58    A1  E8 000CA           BLBS    88(CLUDESC), 10$                        ; 2051
                    0C         A5        04  88 000CE       BISB2   #4, 12(R5)                             ; 2052
20 00000000G        00                   05  E1 000D2 10$:   BBC     #5, LNK$GL_CTLMSK, 11$                  ; 2053
18 00000000G        00                   01  E0 000DA        BBS     #1, LNK$GL_CTLMSK+1, 11$                ; 2054
                    7E         0A    A5  3C 000E2           MOVZWL  10(R5), -(SP)                           ; 2056
                                     55  DD 000E6           PUSHL   R5
7E                  0C         AE        04  C1 000E8       ADDL3   #4, SYMDSCNAM, -(SP)                    ; 2055
                          00000000G  00  9F 000ED           PUSHAB  LNK$AL_SYTBLFMT
                    00000000G  00        04  FB 000F3       CALLS   #4, CRF$INSRTKEY                        ; 2056
                                   0275  31 000FA 11$:       BRW     44$                                     ; 2057
                    55                 01  D0 000FD 12$:   MOVL    #1, FIRST_ABS                           ; 2059
                          00000000'    EF  9F 00100         PUSHAB  PDL                                     ; 2060
                                 10    AE  9F 00106         PUSHAB  VALUE
                              F258     CF     02  FB 00109  CALLS   #2, POP
                                       46     50  E9 0010E  BLBC    R0, 14$
                                       50 00000000'  EF  D0 00111  MOVL    SHRIMGSYM, R0                   ; 2062
                                              1F  13 00118  BEQL    13$
                                              55  D4 0011A  CLRL    FIRST_ABS                           ; 2065
                                       16 00000000'  EF  E8 0011C  BLBS    STAPX, 13$                      ; 2066
11                  0B         A0        06  E1 00123       BBC     #6, 11(R0), 13$                         ; 2067
0C                  0A         A0        03  E1 00128       BBC     #3, 10(R0), 13$                         ; 2068
                    0C         AE    10  A0  D0 0012D       MOVL    16(R0), VALUE                           ; 2071
                    00000000'  EF        50  D0 00132       MOVL    R0, SHRSYMEXPR                          ; 2072
                               3A             5B  91 00139 13$:   CMPB    COMMANDTYPE, #58                ; 2075
                                              25  13 0013C  BEQL    16$
                                       3B             5B  91 0013E  CMPB    COMMANDTYPE, #59                ; 2076
                                              20  13 00141  BEQL    16$
                                    42     8F         5B  91 00143  CMPB    COMMANDTYPE, #66                ; 2077
                                              7D  13 00147  BEQL    19$
                                       56             01  D0 00149  MOVL    #1, SECOND_ABS                  ; 2079
                                             10    AE  9F 0014C  PUSHAB  PDL2                            ; 2080
                                             18    AE  9F 0014F  PUSHAB  VALUE2
                                   F20F      CF        02  FB 00152  CALLS   #2, POP
                                       03                50  E8 00157 14$:   BLBS    R0, 15$
                                                      0234  31 0015A  BRW     47$
                                          00000000'    EF  D5 0015D 15$:   TSTL    SHRIMGSYM               ; 2082
                                                       61  13 00163 16$:   BEQL    19$
                                          00000000'    EF  D5 00165        TSTL    SHRSYMEXPR              ; 2085
                                                       25  13 0016B        BEQL    17$
7E 00000000G        00                               14  C1 0016D        ADDL3   #20, LNK$GL_CURFIL, -(SP)   ; 2088
                          00000000G  00  DD 00175           PUSHL   LNK$GL_RECORD
7E 00000000'        EF                   28  C1 0017B       ADDL3   #40, OBMODESC, -(SP)                    ; 2087
                                         03  DD 00183       PUSHL   #3                                      ; 2088
                          00000000G  8F  DD 00185           PUSHL   #LIN$_ILLSHRXPR
                    00000000G  00        05  FB 0018B       CALLS   #5, LIB$SIGNAL
                                       2B 00000000'  EF  E8 00192 17$:   BLBS    STAPX, 18$              ; 2089
                                       50 00000000'  EF  D0 00199        MOVL    SHRIMGSYM, R0           ; 2090
1F                  0B         A0        06  E1 001A0       BBC     #6, 11(R0), 18$
                                       50 00000000'  EF  D0 001A5  MOVL    SHRIMGSYM, R0                   ; 2091
13                  0A         A0        03  E1 001AC       BBC     #3, 10(R0), 18$
                                       50 00000000'  EF  D0 001B1  MOVL    SHRIMGSYM, R0                   ; 2094
                          14   AE    10  A0  D0 001B8       MOVL    16(R0), VALUE2
                    00000000'  EF        50  D0 001BD       MOVL    R0, SHRSYMEXPR                          ; 2095
                                             56  D4 001C4 18$:   CLRL    SECOND_ABS                      ; 2097
                                          00000000'    EF  D5 001C6 19$:   TSTL    SHRSYMEXPR              ; 2101
                                                       2F  13 001CC        BEQL    20$
                                       33                  5B  91 001CE        CMPB    COMMANDTYPE, #51     ; 2102
```

```
                                                 2A  13 001D1          BEQL    20$                                                 ;
                                          34     5B  91 001D3          CMPB    COMMANDTYPE, #52                                    ; 2103
                                                 25  13 001D6          BEQL    20$                                                 ;
                                7E 00000000G     00  14  C1 001D8      ADDL3   #20, LNK$GL_CURFIL, -(SP)                          ; 2105
                                          00000000G  00  DD 001E0      PUSHL   LNK$GL_RECORD                                       ;
                                7E 00000000'     EF  28  C1 001E6      ADDL3   #40, OBMODESC, -(SP)                                ; 2104
                                                 03  DD 001EE          PUSHL   #3                                                  ; 2105
                                          00000000G  8F  DD 001F0      PUSHL   #LIN$_ILLSHROPR                                     ;
                                   00000000G     00  05  FB 001F6      CALLS   #5, LIB$SIGNAL                                      ;
                                                 50  56  D2 001FD 20$:  MCOML   SECOND_ABS, R0                                    ; 2106
                                   50            55  50  CB 00200      BICL3   R0, FIRST_ABS, R0                                   ;
                                                 50  D2 00204          MCOML   R0, EITHER_SHR                                      ;
                                   0F            33  5B  8F 00207      CASEB   COMMANDTYPE, #51, #15                               ; 2110
                0086            0049          0020      0020B 21$:     .WORD   22$-21$,-                                           ;
                00A2            009B          00B8      00213                  25$-21$,-                                           ;
                00CB            016C          007F      0021B                  30$-21$,-                                           ;
                016C            00D5          00C2      00223                  31$-21$,-                                           ;
                                                                               34$-21$,-                                           ;
                                                                               32$-21$,-                                           ;
                                                                               33$-21$,-                                           ;
                                                                               28$-21$,-                                           ;
                                                                               29$-21$,-                                           ;
                                                                               45$-21$,-                                           ;
                                                                               37$-21$,-                                           ;
                                                                               45$-21$,-                                           ;
                                                                               35$-21$,-                                           ;
                                                                               39$-21$,-                                           ;
                                                                               45$-21$,-                                           ;
                                                                               41$-21$                                             ;
                                          16     50  E9 0022B 22$:     BLBC    EITHER_SHR, 24$                                     ; 2114
                                          08     55  E9 0022E          BLBC    FIRST_ABS, 23$                                      ; 2117
                                00000000'  EF     0C  AE  C0 00231      ADDL2   VALUE, ABSEXPR                                      ; 2119
                                          08     56  E9 00239 23$:     BLBC    SECOND_ABS, 24$                                     ; 2120
                                00000000'  EF     14  AE  C0 0023C      ADDL2   VALUE2, ABSEXPR                                     ; 2122
                                   0C      AE     14  AE  C0 00244 24$: ADDL2   VALUE2, VALUE                                       ; 2124
                                00000000'  EF     10  AE  C0 00249      ADDL2   PDL2, PDL                                           ; 2125
                                                0080  31  00251          BRW     36$                                                 ; 2113
                                          16     50  E9 00254 25$:     BLBC    EITHER_SHR, 27$                                     ; 2130
                                          08     55  E9 00257          BLBC    FIRST_ABS, 26$                                      ; 2133
                                00000000'  EF     0C  AE  C2 0025A      SUBL2   VALUE, ABSEXPR                                      ; 2135
                                          08     56  E9 00262 26$:     BLBC    SECOND_ABS, 27$                                     ; 2136
                                00000000'  EF     14  AE  C2 00265      SUBL2   VALUE2, ABSEXPR                                     ; 2138
                   0C   AE          14     AE     0C  AE  C3 0026D 27$: SUBL3   VALUE, VALUE2, VALUE                               ; 2140
         00000000'  EF          10     AE  00000000'  EF  C3 00274      SUBL3   PDL, PDL2, PDL                                      ; 2141
                                                 5A  11 00281          BRB     38$                                                 ; 2129
                                   0C     AE     0C  AE  CE 00283 28$: MNEGL   VALUE, VALUE                                        ; 2146
                                                 53  11 00288          BRB     38$                                                 ; 2145
                                   0C     AE     0C  AE  D2 0028A 29$: MCOML   VALUE, VALUE                                        ; 2151
                                                 4C  11 0028F          BRB     38$                                                 ; 2150
                                   0C     AE     14  AE  C4 00291 30$: MULL2   VALUE2, VALUE                                       ; 2156
                                                 45  11 00296          BRB     38$                                                 ; 2155
                                                 0C  AE  D5 00298 31$: TSTL    VALUE                                               ; 2161
                                                 40  13 0029B          BEQL    38$                                                 ;
                   0C   AE          14     AE     0C  AE  C7 0029D      DIVL3   VALUE, VALUE2, VALUE                               ; 2162
                                                 37  11 002A4          BRB     38$                                                 ; 2160
                                   0C     AE     14  AE  C8 002A6 32$: BISL2   VALUE2, VALUE                                       ; 2167
                                                 30  11 002AB          BRB     38$                                                 ; 2166
```

```
                51      0C      AE      14      AE  C9 002AD 33$:   BISL3   VALUE2, VALUE, R1                       ; 2172
                50      14      AE              D2 002B3         MCOML   VALUE2, R0                              ; 2173
        50      0C      AE                      CB 002B7         BICL3   R0, VALUE, R0                          ;
        0C      AE              51              CB 002BC         BICL3   R0, R1, VALUE                          ;
                                                1A 11 002C1      BRB     38$                                     ; 2171
                50      14      AE              D2 002C3 34$:    MCOML   VALUE2, R0                              ; 2178
                0C      AE                      CA 002C7         BICL2   R0, VALUE                               ;
                                                10 11 002CB      BRB     38$                                     ; 2177
        0C      AE      14      AE      0C      AE  9C 002CD 35$:   ROTL    VALUE, VALUE2, VALUE                    ; 2183
                                                07 11 002D4 36$:    BRB     38$                                 ; 2182
        0C      AE      14      AE      0C      AE  78 002D6 37$:   ASHL    VALUE, VALUE2, VALUE                    ; 2188
                                        009E    31 002DD 38$:    BRW     46$                                     ; 2187
                        01              0C      AE  D1 002E0 39$:   CMPL    VALUE, #1                           ; 2193
                                                03  12 002E4         BNEQ    40$                                 ;
                                        0089    31 002E6         BRW     44$                                     ;
                        0C      AE      14      AE  D0 002E9 40$:   MOVL    VALUE2, VALUE                       ; 2195
                00000000'       EF      10      AE  D0 002EE         MOVL    PDL2, PDL                           ; 2196
                                        10      AE  9F 002F6         PUSHAB  PDL2                                ; 2197
                                        18      AE  9F 002F9         PUSHAB  VALUE2                              ;
                        F065    CF              02  FB 002FC         CALLS   #2, POP                             ;
                                D9              50  E8 00301         BLBS    R0, 38$                             ;
                                                008A 31 00304        BRW     47$                                 ; 2198
                                59              01  D0 00307 41$:   MOVL    #1, DATALNG                         ; 2206
                50                      5A      59  C1 0030A         ADDL3   DATALNG, COMMANDATA, R0             ; 2207
                00000000'       EF              50  D1 0030E         CMPL    R0, ENDRECADDR                      ;
                                                07  1B 00315         BLEQU   43$                                 ;
                        EDBC    CF              00  FB 00317 42$:   CALLS   #0, REPORT_TIRLNG                   ; 2208
                                                75  11 0031C         BRB     48$                                 ;
                                55              6A  9A 0031E 43$:   MOVZBL  (COMMANDATA), LITINDEX              ; 2209
                                        18      AE  9F 00321         PUSHAB  ENTRY                               ; 2210
                                    F0A9        CF  9F 00324         PUSHAB  ALLOC_LIT                           ;
                                    F095        CF  9F 00328         PUSHAB  COMPARE_LIT                         ;
                                        0C      AE  D4 0032C         CLRL    12(SP)                              ; 2211
                                        0C      AE  9F 0032F         PUSHAB  12(SP)                              ;
                                                55  DD 00332         PUSHL   LITINDEX                            ; 2210
                                00000000'       EF  9F 00334         PUSHAB  LNK$GL_LITBL                        ;
                    00000000G   00              06  FB 0033A         CALLS   #6, LIB$INSERT_TREE                 ;
                                50          18      AE  D0 00341         MOVL    ENTRY, R0                       ; 2212
        0A      A0          07          00  00000000' EF  F0 00345         INSV    PDL, #0, #7, 10(R0)             ;
        0A      A0          01          07  00000000' EF  F0 0034F         INSV    STAPX, #7, #1, 10(R0)           ; 2213
                        0B      A0              55  90 00359         MOVB    LITINDEX, 11(R0)                    ; 2214
                        0C      A0  00000000'   EF  7D 0035D         MOVQ    SHRIMGSYM, 12(R0)                   ; 2215
                        14      A0      0C      AE  D0 00365         MOVL    VALUE, 20(R0)                       ; 2217
                    00 00000000'    EF              55  E2 0036A         BBSS    LITINDEX, LNK$GL_LITFLGS, 44$   ; 2218
                                50                  01  D0 00372 44$:   MOVL    #1, R0                          ; 2219
                                                    1C  11 00375         BRB     48$                             ;
                            EEC2    CF              00  FB 00377 45$:   CALLS   #0, REPORT_ILLTIR2              ; 2222
                                                    15  11 0037C         BRB     48$                             ;
                                    00000000'       EF  DD 0037E 46$:   PUSHL   PDL                             ; 2225
                                7E                  01  CE 00384         MNEGL   #1, -(SP)                       ;
                                            14      AE  DD 00387         PUSHL   VALUE                           ;
                            EF23    CF              03  FB 0038A         CALLS   #3, PUSH                        ;
                                                    02  11 0038F         BRB     48$                             ;
                                                    50  D4 00391 47$:   CLRL    R0                              ; 2227
                                5E                  1C  C0 00393 48$:   ADDL2   #28, SP                         ;
                                                        05  00396       RSB                                     ;
```

; Routine Size: 919 bytes,   Routine Base: $CODE$ + 0F9D

```
;  1667          2228  1 routine compare_dli (value,node) =
;  1668          2229  2 begin
;  1669          2230  2 !
;  1670          2231  2 ! COMPARE A VALUE WITH CURRENT NODE
;  1671          2232  2 !
;  1672          2233  2 map
;  1673          2234  2     node : ref block[,byte];
;  1674          2235  2
;  1675          2236  3 return (.value - .node[dli$l_index])
;  1676          2237  1 end;


                                        0000 00000 COMPARE_DLI:
                                                   .WORD   Save nothing                          ; 2228
                          50       08   AC  D0 00002        MOVL    NODE, R0                     ; 2236
                50        04   AC       0A   A0  C3 00006        SUBL3   10(R0), VALUE, R0
                                                04 0000C        RET                              ; 2237

; Routine Size:  13 bytes,    Routine Base:  $CODE$ + 1334
```

```
;  1678          2238  1 routine alloc_dli (value,retadr) =
;  1679          2239  2 begin
;  1680          2240  2 !
;  1681          2241  2 ! ALLOCATE A NEW DLI BLOCK
;  1682          2242  2 !
;  1683          2243  2 map
;  1684          2244  2     retadr : ref vector[,long];
;  1685          2245  2
;  1686          2246  2 lnk$alloblk(dli$c_size,retadr[0]);
;  1687          2247  2 return true
;  1688          2248  1 end;


                                        0000 00000 ALLOC_DLI:
                                                   .WORD   Save nothing                    ; 2238
                                    08  AC  DD 00002        PUSHL   RETADR                 ; 2246
                                        12  DD 00005        PUSHL   #18                    ;
                          00000000G     00  FB 00007        CALLS   #2, LNK$ALLOBLK        ;
                                        50  01  D0 0000E    MOVL    #1, R0                 ; 2247
                                            04 00011        RET                            ; 2248

; Routine Size:  18 bytes,    Routine Base:  $CODE$ + 1341
```

```
;1690    2249  1 routine controlcommand : tir_proc =
;1691    2250  2 begin
;1692    2251  2 !
;1693    2252  2 ! PROCESS CONTROL COMMANDS
;1694    2253  2 !
;1695    2254  2 external register
;1696    2255  2     commandtype = 11 : byte,
;1697    2256  2     commandata = 10 : ref block[,byte],
;1698    2257  2     datalng = 9;
;1699    2258  2
;1700    2259  2 local
;1701    2260  2     value;
;1702    2261  2
;1703    2262  2 case .commandtype from tir$c_ctl_setrb to tir$c_maxctlcod
;1704    2263  2 of set
;1705    2264  2     [tir$c_ctl_setrb] :                                    ! PROCESS SETTING OF RELOCATION BASE
;1706    2265  3         begin
;1707    2266  3             if (lnk$gl_curpsc = .psectstack[.stackpointer,0]) eql 0     ! SET POINTER TO CURRENT P-SECTION
;1708    2267  4             then begin                                     ! DESCRIPTOR AND IF ZERO
;1709    2268  4                 signal(lin$_notpsect,2,                    ! REPORT AS FATAL ERROR
;1710    2269  4                         obmodesc[omd$b_namlng],lnk$gl_curfil[fdb$q_filename]);
;1711    2270  4                 return false;
;1712    2271  4                 end
;1713    2272  3             else lnk$gl_curisd = .lnk$gl_curpsc[psc$l_isect];          ! SET CURRENT ISD ADDRESS
;1714    2273  3             curpsectbase = .psectstack[.stackpointer,1];               ! SET BASE OF THIS MODULES CONTRIBUT
;1715    2274  3             if not pop (value,pdl)                                     . GIVE UP IF THAT FAILS
;1716    2275  3                 then return false;
;1717    2276  3             stapx = shrimgsym = shrsymexpr = 0;
;1718    2277  3             lnk$gl_locn = .value;                                      ! OTHERWISE SET IT
;1719    2278  2             end;
;1720    2279  2
;1721    2280  2     [tir$c_ctl_augrb] :                                    ! PROCESS AUGMENTS TO RELOCATION BASE
;1722    2281  3         begin
;1723    2282  3             datalng = 4;                                           ! FOUR BYTES OF DATA
;1724    2283  3             if (commandata[longword] + .datalng) gtru .endrecaddr  ! CHECK THE INCREMENT EXISTS
;1725    2284  3                 then return report_tirlng();
;1726    2285  3             lnk$gl_locn = .lnk$gl_locn + .commandata[longword]; ! IF AUGMENT, DO THAT
;1727    2286  2             end;
;1728    2287  2
;1729    2288  2     [tir$c_ctl_dfloc,
;1730    2289  2     tir$c_ctl_stkdl,
;1731    2290  2     tir$c_ctl_stloc] :                                     ! TBK/DBG LOCATION COUNTER CONTROL
;1732    2291  3         begin
;1733    2292  3             local
;1734    2293  3                 entry : ref block[,byte];
;1735    2294  3
;1736    2295  3             if not .lnk$gl_ctlmsk[lnk$v_dbgrec]                    ! MUST BE IN DEBUG TYPE RECORD
;1737    2296  4             then begin
;1738    2297  4                 signal_stop(lin$_illtir,4,                         ! REPORT ILLEGAL TIR TYPE
;1739    2298  4                         .commandtype,obmodesc[omd$b_namlng],
;1740    2299  4                         .lnk$gl_record,lnk$gl_curfil[fdb$q_filename]);
;1741    2300  4                 return false;
;1742    2301  4                 end
;1743    2302  3             else if not pop(value,pdl)                                 ! OTHERWISE POP THE INDEX
;1744    2303  3                 or .pdl neq 0                                          ! AND IF THAT FAILS OR INDEX
;1745    2304  3                 then return false
;1746    2305  4                 else begin                                         !
```

```
;  1747          2306  4                    if .commandtype eql tir$c_ctl_dfloc                          ! IF THIS IS DEFINE LOCATION
;  1748          2307  5                    then begin
;  1749          2308  5                        lib$insert_tree(obmodesc[omd$l_dlilst],.value,           ! INSERT INTO TREE
;  1750          2309  5                                        %ref(0),compare_dli,alloc_dli,entry);
;  1751          2310  5                        entry[dli$l_index] = .value;
;  1752          2311  5                        entry[dli$l_loc] = .lnk$gl_locn;
;  1753          2312  5                        end
;  1754          2313  5                    else begin                                                   ! STKDL OR STLOC
;  1755          2314  5                        if not lib$lookup_tree(obmodesc[omd$l_dlilst],
;  1756          2315  5                                               .value,compare_dli,entry)
;  1757          2316  6                        then begin
;  1758          2317  6                            signal(lin$_illtir,4,.commandtype,obmodesc[omd$b_naming],
;  1759          2318  6                                .lnk$gl_record,lnk$gl_curfil[fdb$q_filename]);
;  1760          2319  6                            return false
;  1761          2320  5                            end;
;  1762          2321  5                        if .commandtype eql tir$c_ctl_stloc
;  1763          2322  5                            then lnk$gl_locn = .entry[dli$l_loc]
;  1764          2323  5                            else return push(.entry[dli$l_loc],-1,0);
;  1765          2324  4                        end;
;  1766          2325  3                    end;
;  1767          2326  2                end;
;  1768          2327  2          [inrange] : return report_illtir2();
;  1769          2328  2        tes;
;  1770          2329  2
;  1771          2330  2 return true
;  1772          2331  1 end;


                                5E              0C  C2 00000 CONTROLCOMMAND:
                                                            SUBL2   #12, SP                              ; 2249
                      04        50    8F              5B  8F 00003          CASEB   COMMANDTYPE, #80, #4   ; 2262
00B0                  00B0            008E          000A     00008 1$:      .WORD   2$-1$,-               ;
                                                    00B0     00010                  5$-1$,-               ;
                                                                                    8$-1$,-               ;
                                                                                    8$-1$,-               ;
                                                                                    8$-1$                 ;
                                50 00000000'    EF  9A 00012 2$:    MOVZBL  STACKPOINTER, R0              ; 2266
                                   00000000'EF40    7F 00019        PUSHAQ  PSECTSTACK[R0]                ;
                          00000000'  EF             9E  D0 00020        MOVL    @(SP)+, LNK$GL_CURPSC     ;
                                                    22  12 00027        BNEQ    4$                        ;
                       7E 00000000G  00             14  C1 00029        ADDL3   #20, LNK$GL_CURFIL, -(SP) ; 2269
                       7E 00000000'  EF             28  C1 00031        ADDL3   #40, OBMODESC, -(SP)      ;
                                                    02  DD 00039        PUSHL   #2                        ;
                                   00000000G    8F      DD 0003B        PUSHL   #LIN$_NOTPSECT            ;
                          00000000G  00             04  FB 00041        CALLS   #4, LIB$SIGNAL            ;
                                                015E    31 00048 3$:    BRW     15$                       ; 2270
                                50 00000000'    EF  D0 0004B 4$:    MOVL    LNK$GL_CURPSC, R0         ; 2272
                          00000000G  00         20  A0  D0 00052        MOVL    32(R0), LNK$GL_CURISD     ;
                                50 00000000'    EF  9A 0005A        MOVZBL  STACKPOINTER, R0          ; 2273
                                   00000000'EF40    7F 00061        PUSHAQ  PSECTSTACK+4[R0]          ;
                          00000000'  EF             9E  D0 00068        MOVL    @(SP)+, CURPSECTBASE      ;
                                   00000000'    EF      9F 0006F        PUSHAB  PDL                       ; 2274
                                            08      AE  9F 00075        PUSHAB  VALUE                     ;
                             EF33       CF             02  FB 00078        CALLS   #2, POP               ;
```

```
                    C8              50  E9 0007D            BLBC    R0, 3$
                            00000000'   EF  7C 00080            CLRQ    SHRIMGSYM                       ; 2276
                            00000000'   EF  94 00086            CLRB    STAPX
    00000000G   00              04      AE  D0 0008C            MOVL    VALUE, LNK$GL_LOCN              ; 2277
                                        1F  11 00094            BRB     7$                              ; 2262
                    59                  04  D0 00096 5$:        MOVL    #4, DATALNG                     ; 2282
50                  5A                  59  C1 00099            ADDL3   DATALNG, COMMANDATA, R0         ; 2283
    00000000'   EF                      50  D1 0009D            CMPL    R0, ENDRECADDR
                                        08  1B 000A4            BLEQU   6$
        EC77    CF                      00  FB 000A6            CALLS   #0, REPORT_TIRLNG               ; 2284
                                  00FD  31  000AB            BRW     16$
    00000000G   00                  6A  C0 000AE 6$:        ADDL2   (COMMANDATA), LNK$GL_LOCN       ; 2285
                                  00EC  31  000B5 7$:        BRW     14$                             ; 2262
                    2B 00000000G    00  E8 000B8 8$:        BLBS    LNK$GL_CTLMSK+2, 10$            ; 2295
7E  00000000G   00                  14  C1 000BF            ADDL3   #20, LNK$GL_CURFIL, -(SP)       ; 2299
                            0000 0000G  00  DD 000C7            PUSHL   LNK$GL_RECORD
7E  00000000'   EF                  28  C1 000CD            ADDL3   #40, OBMODESC, -(SP)            ; 2298
                    7E              5B  9A 000D5            MOVZBL  COMMANDTYPE, -(SP)              ; 2299
                                    04  DD 000D8            PUSHL   #4
                            00000000G   8F  DD 000DA            PUSHL   #LIN$_ILLTIR
    00000000G   00                  06  FB 000E0            CALLS   #6, LIB$STOP
                                  00BF  31  000E7 9$:        BRW     15$                             ; 2300
                            00000000'   EF  9F 000EA 10$:       PUSHAB  PDL                             ; 2302
                                    08  AE  9F 000F0            PUSHAB  VALUE
        EEB8    CF                      02  FB 000F3            CALLS   #2, POP
                    EC                  50  E9 000F8            BLBC    0, 9$
                            00000000'   EF  D5 000FB            TSTL    PDL                             ; 2303
                                        E4  12 00101            BNEQ    9$
50  00000000'   EF                      04  C1 00103            ADDL3   #4, OBMODESC, R0                ; 2308
          52    8F                      5B  91 0010B            CMPB    COMMANDTYPE, #82                ; 2306
                                        30  12 0010F            BNEQ    11$
                                08      AE  9F 00111            PUSHAB  ENTRY                           ; 2308
                              FED6      CF  9F 00114            PUSHAB  ALLOC_DLI
                              FEC5      CF  9F 00118            PUSHAB  COMPARE_DLI
                                0C      AE  D4 0011C            CLRL    12(SP)                          ; 2309
                                0C      AE  9F 0011F            PUSHAB  12(SP)
                                14      AE  DD 00122            PUSHL   VALUE                           ; 2308
                                            50  DD 00125            PUSHL   R0
    000D0000G   00                      06  FB 00127            CALLS   #6, LIB$INSERT_TREE
                    50              08  AE  D0 0012E            MOVL    ENTRY, R0                       ; 2310
        0A          A0              04  AE  D0 00132            MOVL    VALUE, 10(R0)
        0E          A0  00000000G   00  D0 00137            MOVL    LNK$GL_LOCN, 14(R0)             ; 2311
                                        63  11 0013F            BRB     14$                             ; 2306
                                08      AE  9F 00141 11$:       PUSHAB  ENTRY                           ; 2314
                              FE99      CF  9F 00144            PUSHAB  COMPARE_DLI
                                0C      AE  DD 00148            PUSHL   VALUE                           ; 2315
                                            50  DD 0014B            PUSHL   R0                          ; 2314
    00000000G   00                      04  FB 0014D            CALLS   #4, LIB$LOOKUP_TREE
                    2A                  50  E8 00154            BLBS    R0, 12$
7E  00000000G   00                      14  C1 00157            ADDL3   #20, LNK$GL_CURFIL, -(SP)       ; 2318
                            00000000G   00  DD 0015F            PUSHL   LNK$GL_RECORD
7E  00000000'   EF                      28  C1 00165            ADDL3   #40, OBMODESC, -(SP)            ; 2317
                    7E                  5B  9A 0016D            MOVZBL  COMMANDTYPE, -(SP)              ; 2318
                                        04  DD 00170            PUSHL   #4
                            00000000G   8F  DD 00172            PUSHL   #LIN$_ILLTIR
    00000000G   00                      06  FB 00178            CALLS   #6, LIB$SIGNAL
                                        28  11 0017F            BRB     15$                             ; 2319
```

```
                    50      08   AE  D0 00181 12$:    MOVL    ENTRY, R0                       ; 2322
                 53 8F           5B  91 00185          CMPB    COMMANDTYPE, #83                ; 2321
                                 0A  12 00189          BNEQ    13$
          00000000G 00       0E  A0  D0 00188          MOVL    14(R0), LNK$GL_LOCN             ; 2322
                                 0F  11 00193          BRB     14$
                                 7E  D4 00195 13$:     CLRL    -(SP)                           ; 2323
                    7E           01  CE 00197          MNEGL   #1, -(SP)
                                 0E  A0  DD 0019A      PUSHL   14(R0)
                 ED5A CF             03  FB 0019D      CALLS   #3, PUSH
                                 07  11 001A2          BRB     16$
                    50           01  D0 001A4 14$:     MOVL    #1, R0                          ; 2330
                                 02  11 001A7          BRB     16$
                                 50  D4 001A9 15$:     CLRL    R0                              ; 2331
                    5E           0C  C0 001AB 16$:     ADDL2   #12, SP
                                     05 001AE          RSB

; Routine Size:  431 bytes,    Routine Base:  $CODE$ + 1353
```

```
;  1774          2332  1 routine protirdbg =
;  1775          2333  2 begin
;  1776          2334  2 !
;  1777          2335  2 !       PROCESS TIR, DBG AND TRACEBACK RECORDS
;  1778          2336  2 !
;  1779          2337  2 !
;  1780          2338  2 psect plit = $code$;
;  1781          2339  2
;  1782          2340  2 bind
;  1783          2341  2     tirdisptable = uplit(                                  ! TIR DISPATCH TABLE
;  1784          2342  2                          rep tir$c_maxstacod+1             ! STACK   COMMANDS
;  1785          2343  2                              of word(stackcommand-protirdbg),
;  1786          2344  2
;  1787        L 2345  2                          %if tir$c_minstocod-tir$c_maxstacod-1 neq 0
;  1788        U 2346  2                          %then
;  1789        U 2347  2                              rep tir$c_minstocod-tir$c_maxstacod-1       ! ILLEGAL COMMANDS
;  1790        U 2348  2                                  of word(report_illtir-protirdbg),
;  1791          2349  2                          %fi
;  1792          2350  2
;  1793          2351  2                          rep tir$c_maxstocod-tir$c_minstocod+1           ! STORE COMMANDS
;  1794          2352  2                              of word(storecommand-protirdbg),
;  1795          2353  2
;  1796        L 2354  2                          %if tir$c_minoprcod-tir$c_maxstocod-1 neq 0
;  1797          2355  2                          %then
;  1798          2356  2                              rep tir$c_minoprcod-tir$c_maxstocod-1       ! ILLEGAL COMMANDS
;  1799          2357  2                                  of word(report_illtir-protirdbg),
;  1800          2358  2                          %fi
;  1801          2359  2
;  1802          2360  2                          rep tir$c_maxoprcod-tir$c_minoprcod+1           ! OPERATE COMMANDS
;  1803          2361  2                              of word(operatecommand-protirdbg),
;  1804          2362  2
;  1805        L 2363  2                          %if tir$c_minctlcod-tir$c_maxoprcod-1 neq 0
;  1806          2364  2                          %then
;  1807          2365  2                              rep tir$c_minctlcod-tir$c_maxoprcod-1       ! ILLEGAL COMMANDS
;  1808          2366  2                                  of word(report_illtir-protirdbg),
;  1809          2367  2                          %fi
;  1810          2368  2
;  1811          2369  2                          rep tir$c_maxctlcod-tir$c_minctlcod+1           ! CONTROL COMMANDS
;  1812          2370  2                                      of word(controlcommand-protirdbg)
;  1813          2371  2                          )
;  1814          2372  2                                  : vector[,word,signed];
;  1815          2373  2
;  1816          2374  2 psect plit = $plit$;
;  1817          2375  2
;  1818          2376  2 global register
;  1819          2377  2     commandtype = 11 : byte,
;  1820          2378  2     commandata  = 10 : ref block[,byte],
;  1821          2379  2     datalng     =  9;
;  1822          2380  2
;  1823          2381  2 tiroffset  = 1;                                          ! START AT FIRST BYTE PAST RECORD TYPE CODE
;  1824          2382  2 endrecaddr = tirrec[0] + .reclng;                        ! COMPUTE END ADDRESS + 1
;  1825          2383  2 while .tiroffset lssu .reclng
;  1826          2384  3 do begin
;  1827          2385  3     !
;  1828          2386  3     !     STEP THROUGH THE CONCATENATED SEQUENCE OF THE COMMANDS
;  1829          2387  3     !     IN THIS RECORD.
;  1830          2388  3     !
```

```
;  1831          2389  3     if (commandtype = .tirrec[.tiroffset]) gtru 127
;  1832          2390  4     then begin
;  1833          2391  4          !
;  1834          2392  4          ! COMMAND IS A STORE IMMEDIATE
;  1835          2393  4          !
;  1836          2394  4          tiroffset = .tiroffset+1;                              ! POINT PAST THE BYTECOUNT
;  1837          2395  4          datalng   = 256 -.commandtype;
;  1838          2396  4
;  1839          2397  4          if tirrec[.tiroffset + .datalng] gtru .endrecaddr      ! CHECK THERE ARE ARE ENOUGH BYTES I
;  1840          2398  4          then return report_tirlng();
;  1841          2399  4
;  1842          2400  4          if not lnk$wrtimgbyts(.datalng,tirrec[.tiroffset],1)   ! OTHERWISE WRITE THE STREAM
;  1843          2401  4          then return false;
;  1844          2402  4
;  1845          2403  4          tiroffset = .tiroffset + .datalng;                     ! AND UPDATE THE OFFSET INTO THE REC
;  1846          2404  4          end
;  1847          2405  4     else begin
;  1848          2406  4          !
;  1849          2407  4          ! COMMAND IS OTHER THAN STORE IMMEDIATE COMMAND
;  1850          2408  4          !
;  1851          2409  4          commandata = tirrec[.tiroffset+1];                     ! POINT TO DATA PART OF COMMAND
;  1852          2410  4          datalng    = 0;
;  1853          2411  5          if not (if .commandtype gtru tir$c_maxctlcod
;  1854          2412  5                  then report_illtir2()
;  1855          2413  6                  else begin
;  1856          2414  6                       bind routine
;  1857          2415  6                                recordprocessor = .tirdisptable[.commandtype] + protirdbg : tir_proc;
;  1858          2416  6
;  1859          2417  6                       recordprocessor()
;  1860          2418  6                       end
;  1861          2419  5                  )
;  1862          2420  4               then return false
;  1863          2421  4               else tiroffset = .tiroffset + 1 + .datalng;
;  1864          2422  3               end;
;  1865          2423  2     end;
;  1866          2424  2
;  1867          2425  2 return true
;  1868          2426  1 end;


                                                  01502                .BLKB   2
                                        0000* 01504 P.AAB:             .WORD   <STACKCOMMAND-PROTIRDBG>
                                        0000* 01506                    .WORD   <STACKCOMMAND-PROTIRDBG>
                                        0000* 01508                    .WORD   <STACKCOMMAND-PROTIRDBG>
                                        0000* 0150A                    .WORD   <STACKCOMMAND-PROTIRDBG>
                                        0000* 0150C                    .WORD   <STACKCOMMAND-PROTIRDBG>
                                        0000* 0150E                    .WORD   <STACKCOMMAND-PROTIRDBG>
                                        0000* 01510                    .WORD   <STACKCOMMAND-PROTIRDBG>
                                        0000* 01512                    .WORD   <STACKCOMMAND-PROTIRDBG>
                                        0000* 01514                    .WORD   <STACKCOMMAND-PROTIRDBG>
                                        0000* 01516                    .WORD   <STACKCOMMAND-PROTIRDBG>
                                        0000* 01518                    .WORD   <STACKCOMMAND-PROTIRDBG>
                                        0000* 0151A                    .WORD   <STACKCOMMAND-PROTIRDBG>
                                        0000* 0151C                    .WORD   <STACKCOMMAND-PROTIRDBG>
                                        0000* 0151E                    .WORD   <STACKCOMMAND-PROTIRDBG>
                                        0000* 01520                    .WORD   <STACKCOMMAND-PROTIRDBG>
```

```
0000* 01522          .WORD   <STACKCOMMAND-PROTIRDBG>
0000* 01524          .WORD   <STACKCOMMAND-PROTIRDBG>
0000* 01526          .WORD   <STACKCOMMAND-PROTIRDBG>
0000* 01528          .WORD   <STACKCOMMAND-PROTIRDBG>
0000* 0152A          .WORD   <STACKCOMMAND-PROTIRDBG>
0000* 0152C          .WORD   <STORECOMMAND-PROTIRDBG>
0000* 0152E          .WORD   <STORECOMMAND-PROTIRDBG>
0000* 01530          .WORD   <STORECOMMAND-PROTIRDBG>
0000* 01532          .WORD   <STORECOMMAND-PROTIRDBG>
0000* 01534          .WORD   <STORECOMMAND-PROTIRDBG>
0000* 01536          .WORD   <STORECOMMAND-PROTIRDBG>
0000* 01538          .WORD   <STORECOMMAND-PROTIRDBG>
0000* 0153A          .WORD   <STORECOMMAND-PROTIRDBG>
0000* 0153C          .WORD   <STORECOMMAND-PROTIRDBG>
0000* 0153E          .WORD   <STORECOMMAND-PROTIRDBG>
0000* 01540          .WORD   <STORECOMMAND-PROTIRDBG>
0000* 01542          .WORD   <STORECOMMAND-PROTIRDBG>
0000* 01544          .WORD   <STORECOMMAND-PROTIRDBG>
0000* 01546          .WORD   <STORECOMMAND-PROTIRDBG>
0000* 01548          .WORD   <STORECOMMAND-PROTIRDBG>
0000* 0154A          .WORD   <STORECOMMAND-PROTIRDBG>
0000* 0154C          .WORD   <STORECOMMAND-PROTIRDBG>
0000* 0154E          .WORD   <STORECOMMAND-PROTIRDBG>
0000* 01550          .WORD   <STORECOMMAND-PROTIRDBG>
0000* 01552          .WORD   <STORECOMMAND-PROTIRDBG>
0000* 01554          .WORD   <STORECOMMAND-PROTIRDBG>
0000* 01556          .WORD   <STORECOMMAND-PROTIRDBG>
0000* 01558          .WORD   <STORECOMMAND-PROTIRDBG>
0000* 0155A          .WORD   <REPORT_ILLTIR-PROTIRDBG>
0000* 0155C          .WORD   <REPORT_ILLTIR-PROTIRDBG>
0000* 0155E          .WORD   <REPORT_ILLTIR-PROTIRDBG>
0000* 01560          .WORD   <REPORT_ILLTIR-PROTIRDBG>
0000* 01562          .WORD   <REPORT_ILLTIR-PROTIRDBG>
0000* 01564          .WORD   <REPORT_ILLTIR-PROTIRDBG>
0000* 01566          .WORD   <REPORT_ILLTIR-PROTIRDBG>
0000* 01568          .WORD   <OPERATECOMMAND-PROTIRDBG>
0000* 0156A          .WORD   <OPERATECOMMAND-PROTIRDBG>
0000* 0156C          .WORD   <OPERATECOMMAND-PROTIRDBG>
0000* 0156E          .WORD   <OPERATECOMMAND-PROTIRDBG>
0000* 01570          .WORD   <OPERATECOMMAND-PROTIRDBG>
0000* 01572          .WORD   <OPERATECOMMAND-PROTIRDBG>
0000* 01574          .WORD   <OPERATECOMMAND-PROTIRDBG>
0000* 01576          .WORD   <OPERATECOMMAND-PROTIRDBG>
0000* 01578          .WORD   <OPERATECOMMAND-PROTIRDBG>
0000* 0157A          .WORD   <OPERATECOMMAND-PROTIRDBG>
0000* 0157C          .WORD   <OPERATECOMMAND-PROTIRDBG>
0000* 0157E          .WORD   <OPERATECOMMAND-PROTIRDBG>
0000* 01580          .WORD   <OPERATECOMMAND-PROTIRDBG>
0000* 01582          .WORD   <OPERATECOMMAND-PROTIRDBG>
0000* 01584          .WORD   <OPERATECOMMAND-PROTIRDBG>
0000* 01586          .WORD   <OPERATECOMMAND-PROTIRDBG>
0000* 01588          .WORD   <OPERATECOMMAND-PROTIRDBG>
0000* 0158A          .WORD   <REPORT_ILLTIR-PROTIRDBG>
0000* 0158C          .WORD   <REPORT_ILLTIR-PROTIRDBG>
0000* 0158E          .WORD   <REPORT_ILLTIR-PROTIRDBG>
0000* 01590          .WORD   <REPORT_ILLTIR-PROTIRDBG>
0000* 01592          .WORD   <REPORT_ILLTIR-PROTIRDBG>
```

```
                                0000* 01594          .WORD   <REPORT_ILLTIR-PROTIRDBG>
                                0000* 01596          .WORD   <REPORT_ILLTIR-PROTIRDBG>
                                0000* 01598          .WORD   <REPORT_ILLTIR-PROTIRDBG>
                                0000* 0159A          .WORD   <REPORT_ILLTIR-PROTIRDBG>
                                0000* 0159C          .WORD   <REPORT_ILLTIR-PROTIRDBG>
                                0000* 0159E          .WORD   <REPORT_ILLTIR-PROTIRDBG>
                                0000* 015A0          .WORD   <REPORT_ILLTIR-PROTIRDBG>
                                0000* 015A2          .WORD   <REPORT_ILLTIR-PROTIRDBG>
                                0000* 015A4          .WORD   <CONTROLCOMMAND-PROTIRDBG>
                                0000* 015A6          .WORD   <CONTROLCOMMAND-PROTIRDBG>
                                0000* 015A8          .WORD   <CONTROLCOMMAND-PROTIRDBG>
                                0000* 015AA          .WORD   <CONTROLCOMMAND-PROTIRDBG>
                                0000* 015AC          .WORD   <CONTROLCOMMAND-PROTIRDBG>

                                              TIRDISPTABLE=     P.AAB


                                      OFFC 00000 PROTIRDBG:
                                                   .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11                ; 2332
                                53      EAC1   CF  9E 00002          MOVAB   REPORT_TIRLNG, R3
                                52 00000000'   EF  9E 00007          MOVAB   TIROFFSET, R2
                                62             01  D0 0000E          MOVL    #1, TIROFFSET                      ; 2381
                                50          EC  A2  3C 00011          MOVZWL  RECLNG, R0                         ; 2382
                             FC A2          F0 B240 9E 00015          MOVAB   @TIRREC[R0], ENDRECADDR
                                55             62  D0 0001B 1$:      MOVL    TIROFFSET, R5                      ; 2383
                          55    EC  A2         10  00 ED 0001E          CMPZV   #0, #16, RECLNG, R5
                                               76  1B 00024          BLEQU   7$
                                50          F0  A2  D0 00026          MOVL    TIRREC, R0                         ; 2389
                                51                6540 9A 0002A          MOVZBL  (R5)[R0], R1
                                5B                51  90 0002E          MOVB    R1, COMMANDTYPE
                             7F 8F                51  91 00031          CMPB    R1, #127
                                                  36  1B 00035          BLEQU   4$
                                                  62  D6 00037          INCL    TIROFFSET                          ; 2394
                                59                5B  9A 00039          MOVZBL  COMMANDTYPE, DATALNG               ; 2395
                          59 00000100 8F          59  C3 0003C          SUBL3   DATALNG, #256, DATALNG
                          51                      62  C1 00044          ADDL3   DATALNG, TIROFFSET, R1             ; 2397
                                                  51  C0 00048          ADDL2   R0, R1
                                FC    A2          51  D1 0004B          CMPL    R1, ENDRECADDR
                                                  04  1B 0004F          BLEQU   2$
                                63                00  FB 00051          CALLS   #0, REPORT_TIRLNG                  ; 2398
                                                  04  00054          RET
                                                  01  DD 00055 2$:      PUSHL   #1                                 ; 2400
                          7E    F0    A2          62  C1 00057          ADDL3   TIROFFSET, TIRREC, -(SP)
                                                  59  DD 0005C          PUSHL   DATALNG
                             000000000G 00        03  FB 0005E          CALLS   #3, LNK$WRTIMGBYTS
                                38                50  E9 00065          BLBC    R0, 8$
                                62                59  C0 00068          ADDL2   DATALNG, TIROFFSET                 ; 2403
                                                  AE  11 0006B 3$:      BRB     1$                                 ; 2389
                                5A          01 A540 9E 0006D 4$:      MOVAB   1(R5)[R0], COMMANDATA              ; 2409
                                                  59  D4 00072          CLRL    DATALNG                            ; 2410
                                55                5B  9A 00074          MOVZBL  COMMANDTYPE, R5                    ; 2411
                             54 8F                55  91 00077          CMPB    R5, #84
                                                  07  1B 0007B          BLEQU   5$
                             0166 C3              00  FB 0007D          CALLS   #0, REPORT_ILLTIR2                 ; 2412
                                                  0B  11 00082          BRB     6$
                                50       FECD CF45  32 00084 5$:      CVTWL   TIRDISPTABLE[R5], R0               ; 2415
                                         1539 C340  16 0008A          JSB     PROTIRDBG[R0]                      ; 2417
```

```
                 0E              50  E9 0008F 6$:    BLBC    R0, 8$                                  ;
                 50              62  59  C1 00092    ADDL3   DATALNG, TIROFFSET, R0                  ; 2421
                 62        01    A0  9E 00096       MOVAB   1(R0), TIROFFSET                        ;
                                 CF  11 0009A       BRB     3$                                      ; 2383
                 50              01  D0 0009C 7$:    MOVL    #1, R0                                  ; 2425
                                     04 0009F       RET                                             ;
                                 50  D4 000A0 8$:    CLRL    R0                                      ; 2426
                                     04 000A2       RET                                             ;

; Routine Size:  163 bytes,    Routine Base:  $CODE$ + 15AE
```

```
;1870          2427  1 routine prohdrs =
;1871          2428  2 begin
;1872          2429  2 !
;1873          2430  2 !       PROCESS HEADER RECORDS AS FOLLOWS:
;1874          2431  2 !
;1875          2432  2 !               IF HEADER TYPE 0 (MAIN MODULE HEADER) CALL THE MAP
;1876          2433  2 !                       ROUTINE THAT DESCRIBES A MODULE.
;1877          2434  2 !
;1878          2435  2 !               IF A LANGUAGE PROCESSOR HEADER TYPE, CALL THE MAP
;1879          2436  2 !                       ROUTINE THAT OUTPUTS THAT DATA.
;1880          2437  2 !
;1881          2438  2 !               IGNORE ALL OTHER SUB-TYPES.
;1882          2439  2 !
;1883          2440  2 if not  .obmodesc [omd$v_mapmod]                          ! IF MODULE NOT FLAGGED FOR MAP
;1884          2441  2 and not .lnk$gl_ctlmsk [lnk$v_long]                       ! AND THIS IS NOT A FULL MAP
;1885          2442  2 then return true;                                        ! FORGET IT
;1886          2443  2
;1887          2444  2 if not .lnk$gl_ctlmsk [lnk$v_mapopn]                     ! ALSO FORGET IT IF
;1888          2445  2 then return true;                                        ! MAP IS NOT OPEN
;1889          2446  2
;1890          2447  2 if  .objrec [obj$b_subtyp] eql obj$c_hdr_mhd             ! IF THIS IS MAIN HEADER
;1891          2448  2 then lnk$mapmhd (.objrec,.obmodesc)                      ! GO MAP SOME INFO
;1892          2449  2 else if  .objrec [obj$b_subtyp] eql obj$c_hdr_lnm        ! OR IF THE COMPILER IDENTIFICATION
;1893          2450  2      then lnk$mapLng (objrec [obj$b_subtyp]+1,.reclng-2); ! HEADER GO MAP THAT DATA
;1894          2451  2
;1895          2452  2 return true                                              ! RETURN SUCCESS
;1896          2453  1 end;

                                  000C 00000 PROHDRS:.WORD   Save R2,R3                                  ;2427
                        53 00000000'  EF  9E 00002         MOVAB   OBMODESC, R3
                        50            63  D0 00009         MOVL    OBMODESC, R0                         ;2440
                07        14          A0  04 E0 0000C      BBS     #4, 20(R0), 1$
                        35 00000000G  00  E9 00011         BLBC    LNK$GL_CTLMSK+1, 3$                  ;2441
                2D 00000000G          00  05 E1 00018 1$:  BBC     #5, LNK$GL_CTLMSK, 3$                ;2444
                        52                08  A3 D0 00020  MOVL    OBJREC, R2                           ;2447
                                          01  A2 95 00024  TSTB    1(R2)
                                              0D 12 00027  BNEQ    2$
                                              50 DD 00029  PUSHL   R0                                   ;2448
                                              52 DD 0002B  PUSHL   R2
                   00000000G  00              02 FB 0002D  CALLS   #2, LNK$MAPMHD
                                              17 11 00034  BRB     3$
                              01          01  A2 91 00036 2$:  CMPB    1(R2), #1                        ;2449
                                              11 12 0003A  BNEQ    3$
                              7E          04  A3 3C 0003C  MOVZWL  RECLNG, -(SP)                       ;2450
                              6E              02 C2 00040  SUBL2   #2, (SP)
                                          02  A2 9F 00043  PUSHAB  2(R2)
                   00000000G  00              02 FB 00046  CALLS   #2, LNK$MAPLNG
                              50              01 D0 0004D 3$.  MOVL    #1, R0                          ;2452
                                                 04 00050  RET                                          ;2453

; Routine Size:  81 bytes,    Routine Base:  $CODE$ + 1651
```

```
;1898    2454  1 routine construct_dmt =
;1899    2455  2 begin
;1900    2456  2 !
;1901    2457  2 !
;1902    2458  2 !     This routine manufactures Debugger Module/Psect Table (DMT)
;1903    2459  2 !     entries.  There is an entry for each object module that
;1904    2460  2 !     contributes to the DST.  The format of the DMT is illustrated
;1905    2461  2 !     below.
;1906    2462  2 !
;1907    2463  2 !     For each object module with a non-0 DST contribution,
;1908    2464  2 !
;1909    2465  2 !     ------------------------------------------------
;1910    2466  2 !     | Offset into DST of this module's contribution  |       (longword)
;1911    2467  2 !     ------------------------------------------------
;1912    2468  2 !     | Length of this module's DST contribution       |       (longword)
;1913    2469  2 !     ------------------------------------------------
;1914    2470  2 !     | Number of non-0-length psects in this module   |       (longword)
;1915    2471  2 !     ------------------------------------------------
;1916    2472  2 !
;1917    2473  2 !        Then, for each of the above psects in this object module,
;1918    2474  2 !
;1919    2475  2 !        ------------------------------------------------
;1920    2476  2 !        | Base of this module's contribution to the psect |     (longword)
;1921    2477  2 !        ------------------------------------------------
;1922    2478  2 !        | Length of this module's cont. to the psect     |     (longword)
;1923    2479  2 !        ------------------------------------------------
;1924    2480  2 !
;1925    2481  2 !
;1926    2482  2 !     The maximum size of the DMT is calculated by multiplying the number
;1927    2483  2 !     of modules by the length of the module part of the DMT (12 bytes),
;1928    2484  2 !     and adding that quantity to the value obtained by multiplying the
;1929    2485  2 !     total number of psects with the length of the psect part (8 bytes).
;1930    2486  2 !     However, the size of the DMT may be adjusted downward in numerous ways.
;1931    2487  2 !     Among those situations requiring a change in size are 1) no psect
;1932    2488  2 !     entries are made for 0-length psect contributions, and 2) no module
;1933    2489  2 !     entries are made for modules with no DST contribution.
;1934    2490  2 !
;1935    2491  2 !     The offset into the DST of a modules DST contribution is obtained
;1936    2492  2 !     from the value of lnk$gl_dstlocn (DST location counter) at the
;1937    2493  2 !     time the first DBG or TBT record is encountered for a module.
;1938    2494  2 !     The length is obtained by subtracting the DST offset of the previous
;1939    2495  2 !     module from the DST offset of the current module's contribution.
;1940    2496  2 !
;1941    2497  2 !     The psect information is available from the MPC blocks, which used to
;1942    2498  2 !     be mostly for psect output in the linker map.
;1943    2499  2 !
;1944    2500  2 local
;1945    2501  2     zero_length_psect,                  ! Logical flag
;1946    2502  2     curomdptr: ref block[,byte];        ! Ptr to start of data for current module
;1947    2503  2
;1948    2504  2 dmtptr[dcm$l_dstoff] = .lnk$gl_dstlocn;                   ! Start of this module's cont. to DST
;1949    2505  2 dmtptr[dcm$w_numpsc] = .obmodesc[omd$w_hipsct] + 1;      ! Initial number of psects for the module
;1950    2506  2 curomdptr = .dmtptr;                                     ! Save ptr to start of data for this objmod
;1951    2507  2
;1952    2508  2 if .prev_dstlen_addr neq 0                               ! If prev_dstlen_addr has been initialized...
;1953    2509  2 then
;1954    2510  3     begin                                                                   !  then compute dstl
```

```
; 1955          2511  3     if (.prev_dstlen_addr = .dmtptr[dcm$l_dstoff] - .(.prev_dstlen_addr - 4)) eql 0     !  on difference of
; 1956          2512  3     then                                                                                  ! If the last module didn't contribu
; 1957          2513  4         begin                                                                             !  the DMT, then adjust lnk$gl_dmtby
; 1958          2514  4         lnk$gl_dmtbytes = .lnk$gl_dmtbytes -                                              !  reset dmtptr and curomdptr back t
; 1959          2515  4                           (.dmtptr - (.prev_dstlen_addr - 4));                        !  position of the previous module e
; 1960          2516  4         dmtptr = .prev_dstlen_addr - 4;
; 1961          2517  4         curomdptr = .dmtptr;
; 1962          2518  3         end;
; 1963          2519  2     end;
; 1964          2520  2
; 1965          2521  2 prev_dstlen_addr = dmtptr[dcm$l_dstlen];                                                 ! Save address of dstlen for current module
; 1966          2522  2
; 1967          2523  2 if (.dmtptr+dcm$c_size) gtr .lnk$gl_dmtbuffer[1]                                         ! If we'll end up past the end of the alloca
; 1968          2524  2 then                                                                                    !  memory, something's amiss -- quit writing
; 1969          2525  3     begin
; 1970          2526  3     dmtptr = 0;                                                                         !  the DMT...
; 1971          2527  3     return true;                                                                        !  and return
; 1972          2528  3     end
; 1973          2529  2 else
; 1974          2530  3     begin                                                                               ! Otherwise, everything's cool...
; 1975          2531  3     dmtptr = .dmtptr + dcm$c_size;                                                      ! Adjust pointer to psect data part
; 1976          2532  3     incr i from 0 to .obmodesc[omd$w_hipsct] do                                         ! Take a peek at each psect in this module..
; 1977          2533  4         begin
; 1978          2534  4         local pscdsc : ref block[,byte];
; 1979          2535  4         pscdsc = fndpscmapent( .i );                                                    ! Set up psect info...
; 1980          2536  4         pscdsc = .pscdsc[pmt$l_pscdes];
; 1981          2537  4         if .dmtptr neq 0
; 1982          2538  4         then
; 1983          2539  5             begin
; 1984          2540  5             local mpcblk : ref block[,byte];
; 1985          2541  5             mpcblk = .pscdsc[psc$l_mpclst];                                             ! Set up MPC info for the psect...
; 1986          2542  5             while .mpcblk neq 0 do
; 1987          2543  6                 begin
; 1988          2544  6                 if .mpcblk[mpc$l_ownomd] eql .obmodesc                              ! Is this psect owned by this objmod
; 1989          2545  6                 then
; 1990          2546  6                     if .mpcblk[mpc$l_length] neq 0                                  ! If it's a non-0-length psect
; 1991          2547  6                     then
; 1992          2548  7                         begin
; 1993          2549  7                         dmtptr[dcp$l_base] = .pscdsc[psc$l_base] +                  !  then write it's base...
; 1994          2550  7                                              .mpcblk[mpc$l_offset];
; 1995          2551  7                         dmtptr[dcp$l_length] = .mpcblk[mpc$l_length];               !  and length to the DMT
; 1996          2552  7                         zero_length_psect = false;                                  ! Yes, it was a valid psect
; 1997          2553  7                         exitloop;                                                   !  (i.e., non-0 length)
; 1998          2554  7                         end
; 1999          2555  6                     else
; 2000          2556  7                         begin
; 2001          2557  7                         curomdptr[dcm$w_numpsc] = .curomdptr[dcm$w_numpsc] - 1; ! 0-length, so lower the
; 2002          2558  7                         zero_length_psect = true;                                   !  count of psects by 1
; 2003          2559  7                         lnk$gl_dmtbytes = .lnk$gl_dmtbytes - dcp$c_size;            !  and adjust lnk$gl_dmtbyte
; 2004          2560  7                         exitloop;                                                   !  accordingly.
; 2005          2561  6                         end;
; 2006          2562  6                 mpcblk = .mpcblk[mpc$l_nxtmpc];
; 2007          2563  5                 end;                                                                ! of WHILE loop
; 2008          2564  5             if (.dmtptr+dcp$c_size) gtr .lnk$gl_dmtbuffer[1]                        ! If we're going to end up past the end
; 2009          2565  5             then                                                                    !  of our allocated space for the DMT,
; 2010          2566  6                 begin                                                           !  then something's wrong -- stop
; 2011          2567  6                 dmtptr = 0;                                                     !  manufacturing it
```

```
;  2012          2568  6                 return true;                                    !  and return
;  2013          2569  6                 end
;  2014          2570  5             else
;  2015          2571  5                 if not .zero_length_psect                   ! If we got a good psect, then
;  2016          2572  5                 then                                        !  move the pointer to the position
;  2017          2573  5                     dmtptr = .dmtptr + dcp$c_size;          !  for the next psect
;  2018          2574  4             end;
;  2019          2575  3         end;                                                ! of INCR loop
;  2020          2576  2     end;
;  2021          2577  2 return true
;  2022          2578  1 end;
```

```
                                03FC 00000 CONSTRUCT_DMT:
                                           .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9                ; 2454
                59 00000000G    00   9E 00002  MOVAB   LNK$GL_DMTBUFFER+4, R9
                58 00000000G    00   9E 00009  MOVAB   LNK$GL_DMTBYTES, R8
                57 00000000'    EF   9E 00010  MOVAB   DMTPTR, R7
                51              67   D0 00017  MOVL    DMTPTR, R1                            ; 2504
                61 00000000G    00   D0 0001A  MOVL    LNK$GL_DSTLOCN, (R1)
                50      FDDC    C7   D0 00021  MOVL    OBMODESC, R0                          ; 2505
                56        12    A0   3C 00026  MOVZWL  18(R0), R6
          08    A1              56   01 A1 0002A  ADDW3   #1, R6, 8(R1)
                53              51   D0 0002F  MOVL    R1, CUROMDPTR                         ; 2506
                50        FF    A7   D0 00032  MOVL    PREV_DSTLEN_ADDR, R0                  ; 2508
                                18   13 00036  BEQL    1$
          60                    61   FC A0 C3 00038  SUBL3   -4(R0), (R1), (R0)             ; 2511
                                11   12 0003D  BNEQ    1$
                52        FC    A0   9E 0003F  MOVAB   -4(R0), R2                            ; 2515
                52              51   C2 00043  SUBL2   R1, R2
                68              52   C0 00046  ADDL2   R2, LNK$GL_DMTBYTES
                67        FC    A0   9E 00049  MOVAB   -4(R0), DMTPTR                        ; 2516
                53              67   D0 0004D  MOVL    DMTPTR, CUROMDPTR                     ; 2517
    FC    A7                    67   04 C1 00050  1$:  ADDL3   #4, DMTPTR, PREV_DSTLEN_ADDR ; 2521
          50                    67   0C C1 00055  ADDL3   #12, DMTPTR, R0                    ; 2523
                69              50   D1 00059  CMPL    R0, LNK$GL_DMTBUFFER+4
                                51   14 0005C  BGTR    7$
                67              0C   C0 0005E  ADDL2   #12, DMTPTR                          ; 2531
                54              01   CE 00061  MNEGL   #1, I                                ; 2532
                                53   11 00064  BRB     9$
                                54   DD 00066  2$:  PUSHL   I                               ; 2535
              EAF9    CF        01   FB 00068  CALLS   #1, FNDPSCMAPENT
                50              60   D0 0006D  MOVL    (PSCDSC), PSCDSC                      ; 2536
                52              67   D0 00070  MOVL    DMTPTR, R2                            ; 2537
                                44   13 00073  BEQL    9$
                51        0C    A0   D0 00075  MOVL    12(PSCDSC), MPCBLK                    ; 2541
                                2C   13 00079  3$:  BEQL    6$                              ; 2542
              FDDC    C7  04    A1   D1 0007B  CMPL    4(MPCBLK), OBMODESC                   ; 2544
                                1F   12 00081  BNEQ    5$
                          0C    A1   D5 00083  TSTL    12(MPCBLK)                            ; 2546
                                0F   13 00086  BEQL    4$
          62      18    A0  08    A1   C1 00088  ADDL3   8(MPCBLK), 24(PSCDSC), (R2)        ; 2550
                  04    A2  0C    A1   D0 0008E  MOVL    12(MPCBLK), 4(R2)                   ; 2551
                                55   D4 00093  CLRL    ZERO_LENGTH_PSECT                     ; 2552
```

```
                                10  11 00095              BRB     6$                                      ; 2548
                          08    A3  B7 00097 4$:          DECW    8(CUROMDPTR)                            ; 2557
                       55       01  D0 0009A              MOVL    #1, ZERO_LENGTH_PSECT                   ; 2558
                       68       08  C2 0009D              SUBL2   #8, LNK$GL_DMTBYTES                     ; 2559
                                05  11 000A0              BRB     6$                                      ; 2556
                       51       61  D0 000A2 5$:          MOVL    (MPCBLK), MPCBLK                        ; 2562
                                D2  11 000A5              BRB     3$                                      ; 2542
                       52       08  C0 000A7 6$:          ADDL2   #8, R2                                  ; 2564
                       69       52  D1 000AA              CMPL    R2, LNK$GL_DMTBUFFER+4                  ;
                                04  15 000AD              BLEQ    8$                                      ;
                                67  D4 000AF 7$:          CLRL    DMTPTR                                  ; 2567
                                0A  11 000B1              BRB     10$                                     ; 2568
                       03       55  E8 000B3 8$:          BLBS    ZERO_LENGTH_PSECT, 9$                   ; 2571
                       67       08  C0 000B6              ADDL2   #8, DMTPTR                              ; 2573
                    A9 54       56  F3 000B9 9$:          AOBLEQ  R6, I, 2$                               ; 2532
                       50       01  D0 000BD 10$:         MOVL    #1, R0                                  ; 2577
                                04  000C0                RET                                             ; 2578

; Routine Size:  193 bytes,    Routine Base:  $CODE$ + 16A2
```

```
;  2024          2579  1 global routine lnk$objpass2=
;  2025          2580  2 begin
;  2026          2581  2 !
;  2027          2582  2 !       MAIN ROUTINE TO READ RECORDS SEQUENTIALLY FROM OBJECT
;  2028          2583  2 !       MODULE FILES AND DISPATCH ON RECORD TYPE
;  2029          2584  2 !
;  2030          2585  2 own
;  2031          2586  2     tirlocn;                                          ! SAVED LOCATION IN IMAGE BINARY
;  2032          2587  2 local
;  2033          2588  2     first_dbg_record,
;  2034          2589  2     prevomd : ref block[,byte],                       ! PREVIOUS OBJ MOD DESCR
;  2035          2590  2     auxfnb : ref block[,byte];
;  2036          2591  2
;  2037          2592  2 stackpointer = stk$c_size;                           ! SET STACK POINTER
;  2038          2593  2
;  2039          2594  2 if .lnk$gl_dmtbuffer[0] neq 0
;  2040          2595  2 then
;  2041          2596  3     begin
;  2042          2597  3     dmtptr = .lnk$gl_dmtbuffer[0];                    ! POINT TO BEGINNING OF TABLE
;  2043          2598  3     prev_dstlen_addr = 0;
;  2044          2599  3     end
;  2045          2600  2 else
;  2046          2601  2     dmtptr = 0;
;  2047          2602  2
;  2048          2603  2 while lnk$nxtobjmod()                               ! WHILE THERE IS ANOTHER OBJECT MODU
;  2049          2604  3 do begin
;  2050          2605  3     first_dbg_record = true;
;  2051          2606  3     obmodesc = .lnk$gl_curomd;
;  2052          2607  3     lnk$gl_curomd[omd$l_dlilst] = 0;                  ! CLEAR DEBUG LIST (USED DURING PASS
;  2053          2608  3     ch$fill(0,lnk$c_nlits/8,lnk$gl_litflgs);          ! UNDEFINE ALL LITERALS
;  2054          2609  3     stapx = shrimgsym = shrsymexpr = 0;               ! INITIALIZE DATA FOR SHR IMAGE CHEC
;  2055          2610  3     while lnk$nxtrec(objrecdesc)                      ! READ THE NEXT RECORD
;  2056          2611  4     do begin
;  2057          2612  4         case .objrec[obj$b_rectyp] from 0 to obj$c_maxrectyp   ! AND DISPATCH ON THE RECORD TYPE
;  2058          2613  4         of set
;  2059          2614  4 !
;  2060          2615  4 ! HEADER RECORDS
;  2061          2616  4 !
;  2062          2617  4             [obj$c_hdr] :
;  2063          2618  5                 begin
;  2064          2619  5                 prohdrs();                            !  PROCESS THE HEADERS
;  2065          2620  5                 if .objrec[obj$b_subtyp] eql obj$c_hdr_lnm  ! IF THIS IS THE CREATOR IDENT RECOR
;  2066          2621  5                     and .obmodesc[omd$v_nobin]            !  OF A MODULE WITHOUT ANY BINARY
;  2067          2622  5                     then exitloop;                        !  THEN ALL DONE WITH MODULE
;  2068          2623  4                 end;
;  2069          2624  4 !
;  2070          2625  4 ! TIR RECORDS
;  2071          2626  4 !
;  2072          2627  4             [obj$c_tir] :
;  2073          2628  4                 if .lnk$gl_ctlmsk[lnk$v_image]        ! IF THE IMAGE FILE IS STILL OPEN
;  2074          2629  5                 then begin
;  2075          2630  5                     lnk$gl_ctlmsk[lnk$v_dbgrec] = false;  ! RESET POSSIBLE DBG
;  2076          2631  5                     lnk$gl_locn = .tirlocn;               ! RECORD FLAG AND LOCATION
;  2077          2632  5                     protirdbg();                          ! COUNTER AND PROCESS IT
;  2078          2633  5                     tirlocn = .lnk$gl_locn;               ! SAVE THE LOCATION COUNTER
;  2079          2634  5                     end
;  2080          2635  4                 else exitloop;                        ! NO IMAGE FILE SO SKIP REST OF THIS
```

```
;  2081          2636  4
;  2082          2637  4    DEBUG AND TRACEBACK RECORDS
;  2083          2638  4
;  2084          2639  4                [obj$c_dbg,obj$c_tbt] :
;  2085          2640  5                            begin
;  2086          2641  5                                if not .lnk$gl_ctlmsk[lnk$v_image]          ! IF NO IMAGE THEN QUIT NOW
;  2087          2642  5                                then
;  2088          2643  5                                    exitloop;
;  2089          2644  6                                if (.lnk$gl_ctlmsk[lnk$v_dbg]               ! IF /DEBUG REQUESTED
;  2090          2645  7                                    or (.lnk$gl_ctlmsk[lnk$v_trace]         !  OR /TRACE REQUESTED
;  2091          2646  6                                    and (.objrec[obj$b_rectyp] eql obj$c_tbt))) ! AND THIS IS A TRACEBACK RECORD
;  2092          2647  5                                    and not .lnk$gl_ctlmsk[lnk$v_sys]       ! AND THIS IS NOT A SYSTEM IMAGE BEI
;  2093          2648  5                                    AND NOT .LNK$GL_CTLMSK[LNK$V_SHR]       ! AND THIS IS NOT A SHAREABLE IMAGE
;  2094          2649  5                                    and not .obmodesc[omd$v_debuger]        ! AND THIS IS NOT THE DEBUGGER MODUL
;  2095          2650  6                                then begin
;  2096          2651  6                                    lnk$gl_ctlmsk[lnk$v_dbgrec] = true;     ! SET THE DEBUG CONTEXT FLAG
;  2097          2652  6
;  2098          2653  6                                    if .first_dbg_record    ! if this is the first dbg record for the module
;  2099          2654  6                                       and
;  2100          2655  6                                      .dmtptr neq 0        !   and there is dmt information to output...
;  2101          2656  6                                    then
;  2102          2657  7                                        begin
;  2103          2658  7                                        first_dbg_record = false;
;  2104          2659  7                                        construct_dmt();            ! then do it
;  2105          2660  6                                        end;
;  2106          2661  6
;  2107          2662  6                                    lnk$gl_locn = .lnk$gl_dstlocn;          ! SWAP LOCATION COUNTERS
;  2108          2663  6                                    protirdbg();                            ! AND GO PROCESS IT
;  2109          2664  6                                    lnk$gl_dstlocn = .lnk$gl_locn;          ! SAVE UPDATED POINTER
;  2110          2665  5                                    end;
;  2111          2666  4                                end;
;  2112          2667  4
;  2113          2668  4    END OF MODULE RECORDS
;  2114          2669  4
;  2115          2670  4                [obj$c_eom,obj$c_eomw] :
;  2116          2671  5                            begin
;  2117          2672  5                                if .stackpointer eql stk$c_size                 ! IF STACKS HAVE BEEN
;  2118          2673  5                                    then exitloop;                              ! COLLAPSED ALL DONE
;  2119          2674  5                                signal(lin$_eomstk,3,                           ! ELSE REPORT THAT STACK
;  2120          2675  5                                        (stk$c_size - .stackpointer),           ! STILL HAS ITEMS ON IT
;  2121          2676  5                                        obmodesc[omd$b_namlng],lnk$gl_curfil[fdb$q_filename]);
;  2122          2677  5                                stackpointer = stk$c_size;                      ! AND RESET IT
;  2123          2678  4                                end;
;  2124          2679  4                [inrange]   :
;  2125          2680  4                            true;                                               ! IGNORE OTHERS
;  2126          2681  4                [outrange]  :
;  2127          2682  4                            signal(lin$_illrectyp,4,                            ! ILLEGAL TYPES
;  2128          2683  4                                    .objrec[obj$b_rectyp],obmodesc[omd$b_namlng],
;  2129          2684  4                                    .lnk$gl_record,lnk$gl_curfil[fdb$q_filename]);
;  2130          2685  4                tes
;  2131          2686  3            end;                                                        ! END OF RECORD LOOP
;  2132          2687  3        lnk$gl_curisd = 0;
;  2133          2688  2        end;                                                            ! END OF MODULES LOOP
;  2134          2689  2
;  2135          2690  2    lnk$gl_curfil = 0;
;  2136          2691  2
;  2137          2692  2    if .dmtptr neq 0 and .lnk$gl_ctlmsk[lnk$v_image]        ! IF WE'RE PRODUCING A DEBUG MODULE/PSECT INFO TABLE
```

```
; 2138     2693  2 then                                          !  THE LENGTH OF THE LAST OBJECT MODULE'S DST CONTRI
; 2139     2694  2     begin
; 2140     2695  2     if (.prev_dstlen_addr = .lnk$gl_dstlocn - .(.prev_dstlen_addr - 4)) eql 0   ! If the last module didn't
; 2141     2696  2     then                                                                        !  any to the DST, then elim
; 2142     2697  2         lnk$gl_dmtbytes = .lnk$gl_dmtbytes -                                    !  by adjusting lnk$gl_dmtby
; 2143     2698  2                           (.dmtptr - (.prev_dstlen_addr - 4));
; 2144     2699  2     end;
; 2145     2700  2 !
; 2146     2701  2 ! LOOP THROUGH THE CLUSTERS AND CROSS REFERENCE ANY SYMBOLS THAT
; 2147     2702  2 ! ARE ON THE SHR LST AND ARE RELOCATABLE (I.E. DID NOT GET BASED
; 2148     2703  2 ! DURING PASS 2)
; 2149     2704  2 !
; 2150     2705  2 if .lnk$gl_ctlmsk[lnk$v_mapopn]                                                  ! IF MAP FILE IS OPEN
; 2151     2706  2     and .lnk$gl_ctlmsk[lnk$v_long]
; 2152     2707  3 then begin
; 2153     2708  3     local
; 2154     2709  3         cluster : ref block[,byte],
; 2155     2710  3         symbol : ref block[,byte],
; 2156     2711  3         symsnb : ref block[,byte];
; 2157     2712  3
; 2158     2713  3     cluster = lnk$gl_clulst;                                                    ! POINT AT START OF LIST
; 2159     2714  3     while (cluster = .cluster[clu$l_nxtclu]) neq 0
; 2160     2715  3     do if .cluster[clu$v_shrimg]                                                ! IF CLUSTER IS NON-BASED SHR IMAGE
; 2161     2716  3         and not .cluster[clu$v_based]
; 2162     2717  4     then begin
; 2163     2718  4         symbol = .cluster[clu$l_shrlst];
; 2164     2719  4         while .symbol neq 0
; 2165     2720  5         do begin
; 2166     2721  5             symsnb = .symbol - .symbol[sym$b_namlng] - snb$c_fxdlen;
; 2167     2722  5             if .symbol[sym$v_crosref]                                           ! IF SYMBOL WAS CROSS REFERENCED
; 2168     2723  5                 then crf$insrtref(lnk$al_valctltb,symbol[sym$l_value],  ! THEN ENTER IN CREF BY VALUE TABLE
; 2169     2724  5                                   symsnb[snb$b_namlng],.symbol[sym$w_flags]);
; 2170     2725  5             symbol = .symbol[sym$l_shrlnk];                                     ! LINK TO NEXT SYMBOL
; 2171     2726  4             end;
; 2172     2727  3         end;
; 2173     2728  2     end;
; 2174     2729  2 return true                                                                     ! END OF PASS 2
; 2175     2730  1 end;


                                                .PSECT  $OWN$,NOEXE,2
                                    00234 TIRLOCN:.BLKB   4


                                                .PSECT  $CODE$,NOWRT,2

                                        0FFC 00000          .ENTRY  LNK$OBJPASS2, Save R2,R3,R4,R5,R6,R7,R8,R9,-; 2579
                                                                    R10,R11
                                5B 00000000G   00  9E 00002         MOVAB   LNK$GL_CURFIL, R11
                                5A 00000000G   00  9E 00009         MOVAB   LNK$GL_LOCN, R10
                                59     FE37    CF  9E 00010         MOVAB   PROTIRDBG, R9
                                58 00000000G   00  9E 00015         MOVAB   LNK$GL_CTLMSK, R8
                                57 00000000'   EF  9E 0001C         MOVAB   OBMODESC, R7
                             11 A7             19  90 00023         MOVB    #25, STACKPOINTER                               ; 2592
```

```
                                50 00000000G   00    D0 00027          MOVL    LNK$GL_DMTBUFFER, R0                    ; 2594
                                               08    13 0002E          BEQL    1$
                          0224  C7             50    D0 00030          MOVL    R0, DMTPTR                              ; 2597
                                         0220  C7    D4 00035          CLRL    PREV_DSTLEN_ADDR                        ; 2598
                                               04    11 00039          BRB     2$                                      ; 2594
                                         0224  C7    D4 0003B 1$:      CLRL    DMTPTR                                  ; 2601
                    00000000G   00               00    FB 0003F 2$:      CALLS   #0, LNK$NXTOBJMOD                       ; 2603
                                03               50    E8 00046          BLBS    R0, 3$
                                           010E  31    00049             BRW     17$
                                56               01    D0 0004C 3$:      MOVL    #1, FIRST_DBG_RECORD                    ; 2605
                                50 00000000G   00    D0 0004F          MOVL    LNK$GL_CURMD, R0                        ; 2606
                                67             50    D0 00056          MOVL    R0, OBMODESC
                                         04    A0    D4 00059          CLRL    4(R0)                                   ; 2607
              20          00     6E             00    2C 0005C          MOVC5   #0, (SP), #0, #32, LNK$GL_LITFLGS       ; 2608
                                   00000000'    EF       00061
                                      F8        A7    7C 00066          CLRQ    SHRIMGSYM                               ; 2609
                                      10        A7    94 00069          CLRB    STAPX
                                      04        A7    9F 0006C 4$:      PUSHAB  OBJRECDESC                              ; 2610
                    00000000G   00               01    FB 0006F          CALLS   #1, LNK$NXTREC
                                50               50    E9 00076          BLBC    R0, 8$
                                52         08    B7    9A 00079          MOVZBL  @OBJREC, R2                             ; 2612
                          07    00               52    8F 0007D          CASEB   R2, #0, #7
              00A5        004B        FFEB           0031     00081 5$:    .WORD   7$-5$,-
              00A5        FFEB        0061           0061     00089                4$-5$,-
                                                                                   9$-5$,-
                                                                                   14$-5$,-
                                                                                   11$-5$,-
                                                                                   11$-5$,-
                                                                                   4$-5$,-
                                                                                   14$-5$
              7E                6B                 14    C1 00091          ADDL3   #20, LNK$GL_CURFIL, -(SP)               ; 2684
                                      00000000G    00    DD 00095          PUSHL   LNK$GL_RECORD
              7E                67               28    C1 00098          ADDL3   #40, OBMODESC, -(SP)                    ; 2683
                                                 52    DD 0009F          PUSHL   R2                                      ; 2684
                                                 04    DD 000A1          PUSHL   #4
                                      00000000G    8F    DD 000A3          PUSHL   #LIN$_ILLRECTYP
                    00000000G   00               06    FB 000A9          CALLS   #6, LIB$SIGNAL
                                               BA    11 000B0 6$:      BRB     4$
                          00A3  C9               00    FB 000B2 7$:      CALLS   #0, PROHDRS                             ; 2619
                                50         08    A7    D0 000B7          MOVL    OBJREC, R0                              ; 2620
                                01         01    A0    91 000BB          CMPB    1(R0), #1
                                               AB    12 000BF          BNEQ    4$
                                50               67    D0 000C1          MOVL    OBMODESC, R0                            ; 2621
              A3          14    A0               01    E1 000C4          BBC     #1, 20(R0), 4$
                                           0085  31    000C9 8$:         BRW     16$                                     ; 2622
                                FA               68    E9 000CC 9$:      BLBC    LNK$GL_CTLMSK, 8$                       ; 2628
                          02    A8               01    8A 000CF          BICB2   #1, LNK$GL_CTLMSK+2                     ; 2630
                                6A         0228  C7    D0 000D3          MOVL    TIRLOCN, LNK$GL_LOCN                    ; 2631
                                69               00    FB 000D8          CALLS   #0, PROTIRDBG                           ; 2632
                          0228  C7               6A    D0 000DB          MOVL    LNK$GL_LOCN, TIRLOCN                    ; 2633
                                               8A    11 000E0 10$:     BRB     4$                                      ; 2628
                                6C               68    E9 000E2 11$:     BLBC    LNK$GL_CTLMSK, 16$                      ; 2641
              0A                68               06    E0 000E5          BBS     #6, LNK$GL_CTLMSK, 12$                  ; 2644
              61          02    A8               02    E1 000E9          BBC     #2, LNK$GL_CTLMSK+2, 15$                ; 2645
                                05               52    91 000EE          CMPB    R2, #5                                  ; 2646
                                               5C    12 000F1          BNEQ    15$
              58                68               03    E0 000F3 12$:     BBS     #3, LNK$GL_CTLMSK, 15$                  ; 2647
```

```
                50              67  D0 000F7          MOVL    OBMODESC, R0                                    ; 2649
50        14    A0              05  E0 000FA          BBS     #5, 20(R0), 15$
            02  A8              01  88 000FF          BISB2   #1, LNK$GL_CTLMSK+2                             ; 2651
                0D              56  E9 00103          BLBC    FIRST_DBG_RECORD, 13$                           ; 2653
                        0224    C7  D5 00106          TSTL    DMTPTR                                          ; 2655
                                07  13 0010A          BEQL    13$
                                56  D4 0010C          CLRL    FIRST_DBG_RECORD                                ; 2658
          00F4  C9              00  FB 0010E          CALLS   #0, CONSTRUCT_DMT                               ; 2659
                6A 00000000G    00  D0 00113 13$:     MOVL    LNK$GL_DSTLOCN, LNK$GL_LOCN                     ; 2662
                69              00  FB 0011A          CALLS   #0, PROTIRDBG                                   ; 2663
    00000000G   00              6A  D0 0011D          MOVL    LNK$GL_LOCN, LNK$GL_DSTLOCN                     ; 2664
                                8A  11 00124          BRB     6$                                              ; 2611
                19        11    A7  91 00126 14$:     CMPB    STACKPOINTER, #25                               ; 2672
                                25  13 0012A          BEQL    16$
7E              6B              14  C1 0012C          ADDL3   #20, LNK$GL_CURFIL, -(SP)                       ; 2676
7E              67              28  C1 00130          ADDL3   #40, OBMODESC, -(SP)
                7E        11    A7  9A 00134          MOVZBL  STACKPOINTER, -(SP)                             ; 2675
6E              19              6E  C3 00138          SUBL3   (SP), #25, (SP)
                                03  DD 0013C          PUSHL   #3                                              ; 2676
                    00000000G   8F  DD 0013E          PUSHL   #LIN$_EOMSTK
    000000000G  00              05  FB 0C144          CALLS   #5, LIB$SIGNAL
            11  A7              19  90 0014B          MOVB    #25, STACKPOINTER                               ; 2677
                                8F  11 0014F 15$:     BRB     10$                                             ; 2611
                    00000000G   00  D4 00151 16$:     CLRL    LNK$GL_CURISD                                   ; 2687
                          FEE5  31  00157             BRW     2$                                              ; 2603
                                6B  D4 0015A 17$:     CLRL    LNK$GL_CURFIL                                   ; 2690
                52      0224    C7  D0 0015C          MOVL    DMTPTR, R2                                      ; 2692
                                20  13 00161          BEQL    18$
                1D              68  E9 00163          BLBC    LNK$GL_CTLMSK, 18$
                51      0220    C7  D0 00166          MOVL    PREV_DSTLEN_ADDR, R1                            ; 2695
                50        FC    A1  9E 0016B          MOVAB   -4(R1), R0
61 00000000G    00              60  C3 0016F          SUBL3   (R0), LNK$GL_DSTLOCN, (R1)
                                0A  12 00177          BNEQ    18$
                50              52  C2 00179          SUBL2   R2, R0                                          ; 2698
    00000000G   00              50  C0 0017C          ADDL2   R0, LNK$GL_DMTBYTES
4C              68              05  E1 00183 18$:     BBC     #5, LNK$GL_CTLMSK, 22$                          ; 2705
                48        01    A8  E9 00187          BLBC    LNK$GL_CTLMSK+1, 22$                            ; 2706
                54 00000000G    00  9E 0018B          MOVAB   LNK$GL_CLULST, CLUSTER                          ; 2713
                54              64  D0 00192 19$:     MOVL    (CLUSTER), CLUSTER                              ; 2714
                                3C  13 00195          BEQL    22$
F6        58    A4              02  E1 00197          BBC     #2, 88(CLUSTER), 19$                            ; 2715
                F2        58    A4  E8 0019C          BLBS    88(CLUSTER), 19$                                ; 2716
                52        30    A4  D0 001A0          MOVL    48(CLUSTER), SYMBOL                             ; 2718
                                EC  13 001A4 20$:     BEQL    19$                                             ; 2719
                50        0F    A2  9A 001A6          MOVZBL  15(SYMBOL), R0                                  ; 2721
50              52              50  C3 001AA          SUBL3   R0, SYMBOL, R0
                53        FB    A0  9E 001AE          MOVAB   -5(R0), SYMSNB
16        0C    A2              01  E1 001B2          BBC     #1, 12(SYMBOL), 21$                             ; 2722
                7E        0A    A2  3C 001B7          MOVZWL  10(SYMBOL), -(SP)                               ; 2724
                          04    A3  9F 001BB          PUSHAB  4(SYMSNB)
                52              DD  9F 001BE          PUSHL   SYMBOL
                    00000000G   00  9F 001C0          PUSHAB  LNK$AL_VALCTLTB                                 ; 2723
    00000000G   00              04  FB 001C6          CALLS   #4, CRF$INSRTRE                                 ; 2724
                52        1C    A2  D0 001CD 21$:     MOVL    28(SYMBOL), SYMBOL                              ; 2725
                                D1  11 001D1          BRB     20$                                             ; 2719
                50              01  D0 001D3 22$:     MOVL    #1, R0                                          ; 2729
                                    04 001D6          RET                                                     ; 2730
```

```
; Routine Size:  471 bytes,    Routine Base:  $CODE$ + 1763

; 2176          2731  0 end eludom


                                                        .EXTRN  LIB$SIGNAL, LIB$STOP

;                               PSECT SUMMARY
;
;       Name                    Bytes                           Attributes
;
;  $GLOBAL$                       72  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $OWN$                         568  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $CODE$                       6458  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $PLIT$                         14  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  .  ABS  .                       0  NOVEC,NOWRT,NORD ,NOEXE,NOSHR,  LCL,  ABS,  CON,NOPIC,ALIGN(0)


;                           Library Statistics
;                                       -------- Symbols --------      Pages       Processing
;       File                            Total    Loaded   Percent      Mapped      Time
;
;  _$255$DUA28:[SYSLIB]LIB.L32;1         18619        97         0        1000        00:01.9
;  _$255$DUA28:[LINKER.OBJ]DATBAS.L32;1    538       110        20          28        00:00.9


;                                   COMMAND QUALIFIERS
;
;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:LNKOBJPS2/OBJ=OBJ$:LNKOBJPS2 MSRC$:LNKOBJPS2/UPDATE=(ENH$:LNKOBJPS2)

; Size:          6220 code + 892 data bytes
; Run Time:        01:47.5
; Elapsed Time:    03:49.8
; Lines/CPU Min:    1524
; Lexemes/CPU-Min: 21030
; Memory Used:  470 pages
; Compilation Complete
```

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

LNKOBJPS2
LIS

LNKOPTION
LIS